Experimental and Modelling Study of the Alkali-Silica–Reaction in Concrete

THÈSE N^{O} 4510 (2009)

PRÉSENTÉE LE 10 NOVEMBRE 2009

À LA FACULTÉ SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE DES MATÉRIAUX DE CONSTRUCTION
PROGRAMME DOCTORAL EN STRUCTURES

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Cyrille DUNANT

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2009

À Elisa mon amour,
et à mes parents
qui voulaient savoir
ce que je faisais.

Abstract (en)

The alkali-silica reaction (ASR) is a durability issue of concrete. The amorphous silica of aggregates reacts with the alkalies present in the cement paste pore solution to form a hydrophilic gel which swells in the presence of moisture. Many mass concrete structures are affected and understanding of the reaction and its development is crucial, notably for dam owners and managers.

Although some parameters affecting the reaction are well understood, such as temperature, others which depend on the concrete mix design, such as aggregate sizes and particle size distribution (PSD) and external parameters such as the applied load have an effect on the development of the reaction which is not as well understood.

To advance the understanding of ASR an experimental programme was put into place to explore some of these factors. In parallel, a modelling platform was designed and implemented to allow the simulation of the reaction at the material microstructure level.

The expansion of affected mortars and concretes had been linked to the damage state of the aggregates by Ben Haha. We could model this effect and reproduce the effect of changing the aggregate sizes. Simple kinetics were implemented in the model with two factors were required to account for changes in the cure conditions and sample sizes.

The expansion due to the reaction has been reported to be anisotropic in the literature with respect to the direction of casting. We could demonstrate this effect in two independent set of experiments.

The overall shape of the expansion curve was found to be related to the fracture of the aggregates and the interactions between them rather than changes in the rate of the chemical reaction.

The effect of restraining stress was found to more complex than previously reported in the literature, as it notably affects the direction of propagation of microcracks in the aggregates and paste. This leads to an acceleration of the damage and expansion for loads above about 5 MPa threshold.

Keywords: ASR Modelling, prediction, XFEM, concrete microstructure, durability, FEM, meshing, damage.

Résumé (fr)

La réaction alcali-granulats (RAG) est un problème de durabilité du béton. La silice amorphe contenue dans certains agrégats réagit avec les alcalins présents dans la solution de pore de la pâte de ciment pour former un gel qui gonfle en présence d'eau. De nombreuses structures en béton massif sont affectées et une meilleure compréhension de la réaction et de son développement est cruciale, notamment pour les propriétaires et gestionnaires de barrages.

Quoique certains des paramètres affectant la réaction soient bien compris tels que la température, d'autres, qui dépendent des paramètres de mélange du béton tels que la taille des agrégats et la granulométrie, et des paramètres externes tels que la charge appliquée ont un effet sur le développement de la réaction qui est moins établi.

Un programme expérimental a été mis en place pour explorer les conséquences de certains de ces facteurs. En parallèle, une plate-forme de modélisation a été conçue et implémentée qui permet la simulation de la réaction au niveau de la microstructure du matériau.

L'expansion de mortiers et bétons affectés par la réaction avait été liée à l'état d'endommagement des agrégats par Ben Haha. Nous avons pu modéliser cet effet et reproduire numériquement les conséquences de la variation des tailles des agrégats. Un modèle cinétique a été développé qui ne requiert que deux facteurs linéaires pour tenir compte de différences en terme de cure et de forme d'échantillons.

le gonflement dû à la réaction a été décrit dans la littérature comme étant anisotrope par rapport au sens du coulage. Nous avons pu confirmer cet effet au cours de deux séries d'expériences.

La forme de la courbe d'expansion a été reliée à la mécanique de la rupture des agrégats et leurs interactions plutôt qu'à des changements dûs à la vitesse de la réaction chimique.

Nous avons établis que l'effet du confinement était plus complexe que ce qui était rapporté dans la littérature. En effet, l'application d'une contrainte oriente la propagation des micro-fissures dans les agrégats et la matrice. Ceci conduit à une accélération et une augmentation de l'expansion dans les directions normales à la contrainte, passé un seuil d'environ 5 MPa.

Mots-clés: RAG, modélisation, prédiction, XFEM, micro-structure du béton, durabilité, éléments finis, maillage, endommagement.

Contents

List of Figures

List of Tables

Introduction

The alkali-silica reaction (ASR) was discovered in the 1940s [3]. Since its discovery, occurrences have been reported nearly everywhere. Although all kind of structures are affected, those made from mass concrete such as dams pose significant challenges. Those structures are designed for very long service lives, and cannot be easily repaired. Further, any repair is very expensive. The lack of a good predictive model for the development of ASR has forced dam owners and supervisors to perform repairs when the structural integrity or functionality of the structure is impaired with no guarantee that reconstruction might not be more cost-effective.

Numerical models to predict the effect of ASR have been proposed by Léger [4] and others using a pseudo-thermal expansion to model the effect of ASR, but ten years later, purely phenomenological parameters still need to be fit such as in the models of Saouma or Multon [5, 6]. Also, those models require extensive experimental programmes because many parameters are required for input.

Constitutive Modelling of ASR ASR is a complex phenomenon, whose manifestations can vary depending on the reactive aggregate considered [7]. The apparent development can be affected by chemical and mechanical parameters [8] which can alter both the kinetics and its manifestation in the material in terms of expansion and damage. The number of parameters affecting the reaction is large, and phenomenological models have so far been only partially successful in predicting the development of the reaction accurately enough for engineering applications.

Understanding the microstructural manifestation of ASR is key to the construction of a predictive model. A physical understanding of the reaction process and of the induced degradation is therefore necessary for the formulation of such a model. In his thesis we have concentrated on mechanical aspects.

The chemical interactions involved in the reaction are complex, although the basic reaction is well understood. The silanox bonds break to form silanol compounds. The subsequent adsorption of water leads to swelling of the gel. Depending on the alkali which was involved in the reaction, gels of different nature can be formed at different rates [9, 10, 11]. But in all cases, a gel is formed at the sites where reactive silica was available, and this gel swells in the presence of moisture.

Therefore, the mechanical consequences of the ASR can be modelled assuming the localised presence of a swelling gel. Previous work by Ben Haha [2, 1] indicates that there is a strong link between a measure of the advancement of the reaction and the macroscopically recorded expansion. The observations by Ben Haha were used as a starting point for the formulation of our micro-mechanical model. From the mechanical perspective the reaction is slow enough that the damage induced by it can be assumed to arise in quasi-static conditions: the only externally imposed state parameter to consider is the amount of gel at each step of the reaction.

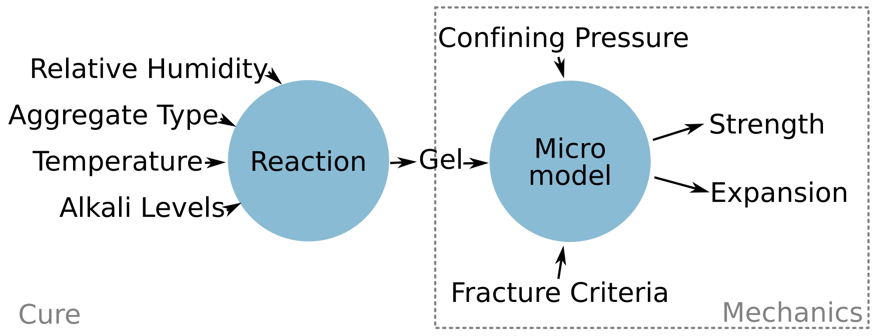


Figure 1: The two aspects of ASR are represented schematically. The chemistry and the inputs which affect the reaction leading to the production of gel on one hand and the microstructural make-up which is affected by the presence of the gel.

Experimental Programme An experimental programme was designed to measure the effects of various conditions on the development of the reaction. The first phase of the experimental program consisted in verifying the generality of the link between reaction and expansion. Mortars were cast and their expansion measured. Samples were prepared at intervals for back-scattered electron (BSE) examination and damage quantification using a custom-developed image analysis (IA) programme. The link previously observed between damage and expansion was confirmed.

A revised protocol was designed to minimise leaching of alkalies over the long term. This experimental design ensures that the mechanical aspects can be measured independently of the chemical aspects, and that the results of the experiments can be used to validate the model.

ASR is a phenomenon fundamentally related to the damage state at the microstructure level. Three aspects of damage evolution are predominant in their effects at the macro-scale: the propagation of cracks in the aggregates and the size effects which are specific to each aggregate size class; the interactions between expanding aggregates of different sizes in a densely packed microstructure; finally, the influence of externally imposed stresses.

To assess the effects of size, two experimental sets of samples were prepared: concretes cast with particle size distribution (PSD)s with different maximum diameter (D_{max}), using only reactive aggregates, and concretes cast with the same PSD but each with a different size class of reactive aggregates. The combination of both experimental setups allowed us to link geometrical factors with kinetic effects.

To evaluate the influence of externally imposed stress, notably the redistribution of strains and the volumic evolution of affected concrete, samples instrumented with internal strain gauges and temperature sensors were prepared and cured in alkaline solutions in modified creep frames. The radial and longitudinal strains were then recorded for four loading conditions, ranging from 0 to 15 MPa, for reactive and non-reactive samples.

As ASR is known to be anisotropic in free conditions[12, 13, 14], a control was required so the free anisotropy could be measured. ASR depends on the casting direction [8], therefore a block of concrete was cored in the directions normal and perpendicular to the casting. The cores were rectified and immersed in alkaline solutions and their expansion recorded. The combination of those two experimental sets allowed the study of the macroscopic redistribution of ASR-induced strains.

ASR Numerical Model To study the link between the formation of the gel and the subsequent expansion and loss of mechanical properties recorded at the macroscopic scale, a combined modelling and experimental approach is required. Extensive experimental studies of the ASR which have been conducted, notably by Poyet and Larive [15, 12]. They notably explored the parameter space of the factors affecting the reaction. But in these studies there was no clear separation of the chemical and mechanical aspects, and the ASR was treated as a global chemo-hydro-thermo-mechanical process.

In this study, leaching was to be prevented by alkalies in the curing baths of the samples. Further experimentation and modelling indicated that the curing conditions caused the reaction rate to be constant. Thus, we could construct and validate a micro-mechanical model of the reaction formulated with no mechanical-chemical coupling. All experimental samples in this study were cured in alkali-loaded solutions so no leaching would occur, no shrinkage would happen and no reactant would become exhausted other than perhaps the amorphous silica in the aggregates. These curing conditions can have different kinetic consequences on the reaction depending on the choice of aggregate, but they ensure that the reaction will occur at a constant rate throughout the experiments. The micro-mechanical model we have constructed could then be validated using the experimental data: the macroscopic consequences recorded in the experiments and those predicted by the model should be identical, given a simple linear transform from the amount of reaction to time (Fig. 1).

The links between the microstructure and the macroscopic expansions are not well understood. We developed a numerical tool which is able to integrate as much of the complexity of the experimental reality so the various parameters can be explicitly simulated in the model. Constructing the numerical model of ASR posed significant challenges: the microstructure of a representative elementary volume (REV) of mortar, let alone concrete, is sufficiently complex that even linear properties from explicitly meshed microstructures are difficult to obtain — because the mesh is difficult to produce. This comes from the inherently multiscale nature of the involved geometries: aggregates with sizes ranging from 0.1 mm in diameter to 16 mm or more. To such an initial setup must be added the ASR gel pockets which are yet two to three orders of magnitude smaller than the smallest aggregate.

To be able to simulate explicitly (in 2D) the complete microstructure, the classical finite element modeling (FEM) approaches was combined with the extended finite element modeling (XFEM). This required the integration of a mesher and the development of a specialised virtual machine (VM), suitable for the implementation of various enrichment schemes. The introduction of mechanical degradation in the microstructure is done with a global damage model which was developed to have good convergence properties with mesh refinement, as well as the ability to reproduce morphological features of damage, such as crack patterns. To integrate all these features, various developments were required: a mesher capable of fully automatic operation and capable of fast computation of mesh-geometry interaction; the damage model needed to produce globally correct energy convergence with highly localised features; a novel integration of damage and XFEM was required; finally, the density of enrichment features in the simulations is much larger than those reported in the literature.

The model developed was then used to simulate the experimental setup. These simulations gave insights on the mechanical interactions at the meso level which could explain some of the experimental observations, notably the role of individual aggregate size and of the D_{max} of the PSD.

Chapter Organisation

Chapter 1 outlines the architecture of the modelling framework developed, with a particular focus on mesher integration.

Chapter 2 presents the damage simulation algorithm developed for the simulation of densely cracked composites.

Chapter 3 first surveys literature on the influence of the curing conditions and mix design on the reaction, then describes the experimental programme put in place and presents the results obtained.

Chapter 4 discusses ASR modelling in the literature and shows the basic application of the model: the simulation of free expansion and the prediction of mechanical properties degradation.

Chapter 5 focuses on the development of a kinetic model and presents simulation results for different aggregate sizes and reactive fractions on one hand, and on the effect of externally imposed loads on the other hand.

Chapter 6 is conclusions and perspectives.

Part I

Numerical developments

CHAPTER 1

AMIE Architecture

Parts of this chapter have been published in the Revue Européenne de Mécanique Numérique (special XFEM *edition).*

1.1 Introduction

The simulation of the alkali-silica reaction (ASR) at the microstructure level poses significant challenges. Concrete itself is a multi-scale material, and simple meshing of the aggregates is difficult, not least because as their number is large, they need to be generated and placed by an algorithm. The phenomenological observation we wished to model numerically involved growing pockets of gel orders of magnitude smaller than the aggregates. In turn, the swelling of the gel induces damage throughout the microstructure.

The simulation of ASR requires complex setups and the concurrent usage of numerical techniques not typically found simultaneously in available codes (extended finite element modeling (XFEM) and damage). The complexity of the simulation setup lies essentially in the interaction between the different numerical approaches. Therefore, using a commercial code would have meant reimplementing most of what makes the core of a finite element framework (elements, behaviour, problem description), and using only such parts as the solvers. The simulations and techniques also require access to a mesher for efficient and scalable implementation.

The particular nature of the problem required many special developments which were better written in a generic framework. As the best numerical tool to achieve our goals was not known at the beginning, the finite element implementation of Automated meshing for integrated experiments (AMIE) was written as a generic implementation of the Galerkin method [16], as nearly all implicit numerical modelling methods are derived from it. The Galerkin method is a generic method for finding least-square approximations to the solutions of partial differential equation problems. This formalism was preferred to the more common finite element modeling (FEM) approximation which assumes Lagrange polynomials to be used for the approximation to the solutions. This assumption is indeed invalid in the case of XFEM of meshless methods. The method was first suggested by Galerkin in 1915 as an approximation method to solve a problem involving pivots and planes [16]. The finite element method, the extended finite element method and spectral methods are all derived from it.

In this chapter, the state of the art of numerical framework is reviewed, then the architecture of AMIE described in more detail, with a special focus on the mesher and the XFEM implementation. A special treatment of damage was also required. This is discussed in the next chapter.

1.2 FEM framework architecture

1.2.1 Developments in FEM Frameworks Architecture

Since the invention of FEM [17, 18] many programmes implementing the FEM have been written, and the architecture of these programmes has evolved with the available computational power. Some are single-purpose codes designed to solve a given problem and some, more generic, attempt to provide functionality to allow a range of applications. The first finite element softwares developed were used for large-scale simulations requiring high performance. In turn, the performance requirement limited the choice of computer languages to those for which good optimising compilers existed. Therefore, until the 80s, the formula translating system (FORTRAN) was the language of choice.

This imposed a way of thinking about the finite element method essentially centred around the discrete form of the problem and the optimisation of the linear algebra operations required for the solving of the systems of equations. In the 80s, when object orientation became a focus of study in computer science, with the implementation of languages such as Smalltalk and C++'s ancestor "C with classes", tools became available to design new types of finite element codes.

As running a simulation involves many steps, each requiring dedicated functionality, frameworks linking the different tools in a single package have been developed around the existing codes.

Monolithic Frameworks

Historically, finite element frameworks were monolithic. They were a single set of functions, built around a highly optimised kernel in FORTRAN which was designed almost only for numerical performance. This kernel provided the numerical operations necessary to solve a specific kind of problem (elastic modelling, fracture mechanics, fluid mechanics).

This approach is falling into disfavour because of the highly complex nature of the kernels and their total lack of flexibility. However, large well-optimised bodies of industrial code exist, some of which have been released as open-source (code ASTER from EDF) or are maintained in-house (the CEA's CAST3M). A more agreeable and productive user-interface is provided using an application programming interface (API) built on scripting languages, frequently Python [19]. This approach offers a good compromise between high efficiency and scalability, provided by the kernel, and ease of use, coming from the high-level API. However developments are limited to concepts which do not touch the kernel [20].

When a required functionality is not available from the core of these programmes, it is obtained by linking to commercial, closed codes, which frequently offer interfaces for plug-ins.

Pluggable Frameworks

Pluggable frameworks provide element libraries and offer a list of typical physical behaviour such as elasticity, plasticity, heat transfer, etc. as well as solvers and post-processors. They also provide the users with high or low-level interfaces to be more extensible. Such frameworks are very popular when special-purpose capabilities are needed with no real constraint on scalability: required functionality is implemented outside of the main loop of the simulation. Many in-house research codes are such developments, built on an existing commercial framework, such as ABAQUS™ [21], or Ansys™. They consist of typically small or large plug-ins used to provide a specific functionality to a framework. This type of development is suitable when for example a new mechanical model is required, or coupled equations have been derived and need to be tested [22, 23]. If a problem, such as the simulation of ASR, requires both damage, which affects the weak form of the elements, and XFEM which affects the nature of the elements simultaneously, the setup of the simulation might not be possible as it would require a modification of the core of the code, as both techniques, XFEM and damage and not usually found together.

Self-Contained Special Purpose Developments

More general developments of numerical techniques have first been introduced as prototype implementations, based on interpreted languages, such as MATLAB. These are typically not produced with scalability in mind: they are proof-of-concept [24, 25], or used as examples [26] to illustrate a particular method.

Alternatively, some special-purpose developments are focused on high performance. They serve as a demonstration that a numerical algorithm is faster or better than the state of the art. These developments focus on the solving of model problems, or provide specialised algorithms useful in the finite element formulation [27, 28]. The problems solved in such cases are provided by specialised benchmark libraries designed to test purely numerical aspects. Other developments focus on solving a particular problem. Little attention is given to the re-usability of the code, and the value of the simulation lies not in the numerical aspects but in the physical implications derived from the solution. The role of a framework is to allow the easy integration of such development efforts, so they can be used in conjunction.

Special purpose developments do not form frameworks, they form model implementations of basic numerical tools or techniques, or extension of frameworks. They are the most frequent examples of research codes. The choice of a framework, or of writing single-purpose programmes is constrained by a series of considerations:

1. Time required for the implementation
2. Range of problems to be solved
3. Availability of existing codes
4. Performance required
5. Skill of the developers
6. Skill of the users

Architectural Developments

Integration of tools, such as geometry libraries, solvers and meshers by providing a unified API for the developer is desirable in terms of development speed but also provides opportunities to leverage synergies between the tools. Some research focuses on the architectural implications of integration. The literature on such developments and the developments themselves have usually a limited scope, and typically only concentrate on a single aspect of the simulation: mesh generation, solvers, or the finite element matrix generation. This is due to material constraints and lack of developer time. However such developments then cannot use the synergies possible from having for example both mesh generation and finite element available in the same set of libraries. Indeed, the generation of mesh and geometry is time consuming and critical for the good performance of the solvers, and integrated post-processing is necessary for some complex setups. There are therefore two approaches to integration: integration to an existing body of functionality or the development of complete packages.

Integration Commercial packages frequently offer integrated solutions, which can interface which computer-aided design (CAD) tools for geometry and meshing. They further provide display and post-processing tools. The integration of finite element modelling within such a larger framework is not a new concept, as Fredriksson and colleagues in 1981 already discussed the integration between CAD software and finite element analysis [29]. The integration was meant to dramatically reduce the time needed to optimise designs. The research on user interaction with these programmes is centred more on data exchange and visualisation, and assumes a framework for the computation exists [30, 31]. Others focus purely on user interaction, for example Mackie describes how a framework can be made interactive for users [32]. Central to integration with an existing framework is data exchange. Hughes and colleagues [33] have proposed a new finite element type, using the non-uniform rational B-spline (NURBS) as shape functions because CAD software describes geometry as NURBS so integration between design and analysis is facilitated. There are also excellent numerical reasons for such a proposition: the error on the result is reduced with the increasing order of the polynomial used for the approximation. An integrated framework centred on the algorithms and tools has been proposed by Rucki and colleagues [34]. The framework they proposed was designed to allow easy switching of algorithms and is centred around the concept of *coordinate-independent data types*[1] to describe the elements. Although the method is completely different, the goal is the same as in [33]: the integrated finite element part must exploit the geometry which is determined externally. Yu and colleagues emphasise the need for flexible element types, with separate geometry and interpolation method, but do not tackle the problem of obtaining the geometry [35]. These examples highlight a constant in the development of

[1]It is convenient to describe geometry using global coordinates, but behaviours in elements are best represented using local coordinates.

frameworks: there is a geometrical part to the formulation of the problem, and finite element "frameworks" frequently assume the existence of a mesh. Coordinate independence is obtained in the programme developed here, AMIE, from the use of coordinate transformation functions generated on-the-fly, and algorithms expressed only in the local coordinate systems of the elements. This property did not require any design choice and came as a consequence of the availability of a virtual machine. AMIE also provides its own set of geometry routines.

Framework Architecture There are also examples of research focused on the interaction between the different parts of a framework. Much original work in finite element software architecture was centred on the linear algebra aspect. With an `Element` class central to the design, Dubois and Zimmermann have their `Element` class serving essentially as an interface for the computation of the elementary matrix [36, 37]. These architectural advances led to the separation of the finite element formulation and the solver implementation.

This allows the modification of a single part of the code without requiring extensive modifications throughout. This requirement is well-met using the object-oriented programming paradigm [38]. However, when large scale problems need to be solved, careful implementation is required to get good performance. Besson and colleagues proposed an interface which uses the data encapsulation possibilities offered by C++ to implement a framework geared towards large scale problems [39]. More recently, with the advent of meshless methods[2] [40] and the multiplication of numerical methods derived from the Galerkin approach [16], the "elements" became collections of nodes with associated basis functions, distinct from the "elements" of a mesh. The architecture, however did not fundamentally change, and developments are still centred on the separation between elementary matrix generation and solving the global system. The requirement for many different and often antagonistic features prompted the development of a complete framework, AMIE.

1.3 AMIE Architecture

1.3.1 Overview

In Amie, we have investigated the role of the mesher in a framework and the synergies which could be obtained [41]. This point is further developed

[2]Meshless methods typically still depend on the presence of a mesh, but the topological mesh structure is not reflected in the assembly of the problem matrix.

in Chap. 1.4.3. Only the solvers were not requiring special-purpose developments, and they represent a relatively small effort in terms of development compared to the design of the architecture. Fully automatic meshing and mesh-geometry interaction computation are other useful features which are provided within the AMIE framework. They then can be employed as needed, for example for the automatic generation of quadratures in enriched elements.

AMIE was designed so it is possible to describe a problem using the available tools from the library and let the framework compute a solution. Then, using the post-processing tools also provided, the result can be displayed externally. The framework was designed to have a separation between the mathematical tools, such as geometry and linear algebra used in the different steps of a computation. Specific programming interfaces are provided for each task and reflect the usual notation conventions. This is different than good object-orientation which only requires encapsulation of the various concepts. Indeed, AMIE is not architectured around the finite elements as is common, but rather around the mesher, virtual machine and geometry library. Mechanical behaviours, elements and extended finite elements can then be implemented as needed very easily.

The functions provided by AMIE are both high-level and low-level. The high level functions are useful to simply describe the geometry and use the already designed behaviour laws. Using these functions does not require advanced knowledge of numerics. The low-level functionality offers programming hooks to implement new element types, behaviour or quadrature schemes for example.

The expression of the mathematical problem as a weak form[3] is separated from the computational geometry required for the mesher. The solvers, involve mostly linear algebra form a third group of functionalities. Thus groups of functionality are bound together by types of techniques, and reflect the mathematical tools and techniques best suited to tackle each sub-task. Within each group, a more classical object oriented approach is used, bundling data and methods in a structured way. In order to provide the user with an easy-to-use programming interface which links the various parts of the framework a *problem descriptor* was implemented. The main elements of AMIE are:

- The problem descriptor
- The meshers
- The behaviours

[3]The weak form of a differential equation is an integral form. It is *weak* because it uses a less restrictive solution space.

- The computational geometry module
- The element library
- The virtual machine
- The solvers

Together they form a complete framework, which can be extended to any type of finite element modelling. When following an object-oriented design approach [42], the facilities used by each object to perform its task are available for the others to use. The geometrical data and boundary conditions are provided by the user through the interface of the program itself. The mesher provides the geometry of the elements, and the neighbourhood relationships. The assembled matrices are the responsibility of an assembler, and form a global matrix which is then passed to a solver. Figure 1.1 illustrates the interactions and data flows between the different elements of the program, notably these enabled by the mesher.

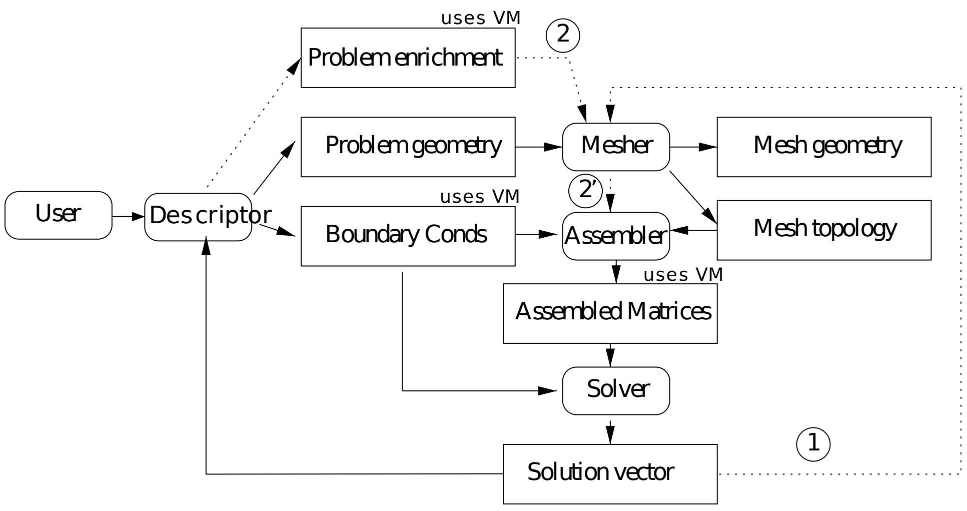


Figure 1.1: The additional data paths made possible by the integration of the mesher: 1, for adaptive meshing, and 2 and 2′ for computationally efficient enrichment

Problem descriptor

The so-called *problem descriptor* allows the user to input the geometry, the behaviour laws and the precision desired. The problem descriptor binds together the various parts of AMIE. It is designed to allow a high-level description of problems: for example, a mortar bar, with a given aggregate size distribution, and some behaviour for the various material phase present.

It is based on objects such as `Sample`, `Inclusion`, `Pore`, `Crack`, etc. In turn, those have a member `behaviour`, such as `LinearElastic`.

In AMIE, a design choice was to provide the user with an API for the description and generation of problems, rather than an input file format of a graphical user interface (GUI). Such modes of input could be written on top of, or using the API. Other finite element frameworks, such as OOFEM are entirely defined around the input file format, as all classes are serialised[4]. This is restrictive because some setups cannot be practically described by hand, such as the explicit microstructure of a mortar bar. These are however easily described using a rule for the generation of appropriately sized aggregates and a placement algorithm, both provided by AMIE. Therefore, the input file needs to be produced by a separate programme in any case.

Computational Geometry

The core task in describing a problem, is its geometry; the position of aggregates, pores and notches. Frequently, geometry and drawing of meshes is an external task from the point of view of the core finite element code. However, we wished to be able to perform various geometry-related tasks which would be central to the problems at hand. In the case of concrete microstructure, the geometry can be easily represented by a large number of simple primitives. For example, generation from a particle size distribution involves the creation of a list of virtual aggregates, with their radii, but it also involves placing the aggregates in a sample. The latter task requires geometrical intersection tests. Another usage for geometry is for the definition of XFEM enrichments which are typically functions based on a geometrical support, a particle or a crack for example. Finally, and also related to the XFEM implementation are mesh-geometry interaction computations: a mesh is composed of geometrical elements, and the interactions only need to be defined in the geometry library. It was therefore appropriate to integrate a geometry library in AMIE to provide the needed functions.

In general, geometrical objects can be separated into two categories:

1. simple objects described as analytic functions (circles, spheres),
2. general shapes (Bézier patches, Boolean trees of primitives).

[4]Serialisation is a programming technique through which data is transformed in such a way that it can be saved and read as a stream.

The computational geometry module provides most of the usual shapes (spheres, circles, lines, points. . .) and their interactions. All those objects are abstracted to a purely virtual interface.

The geometry library is based on the principle of constructive solid geometry: complex shapes are defined from Boolean operations between geometry primitives. The library defines the primitives and provides tests for the intersections. Fig. 1.2 illustrates the construction of a complex-shaped pore using a tree of Boolean operations on geometrical shapes. The API makes no assumption on the dimensionality of the described objects and therefore makes it uniquely suited to represent such concepts as level sets [43].

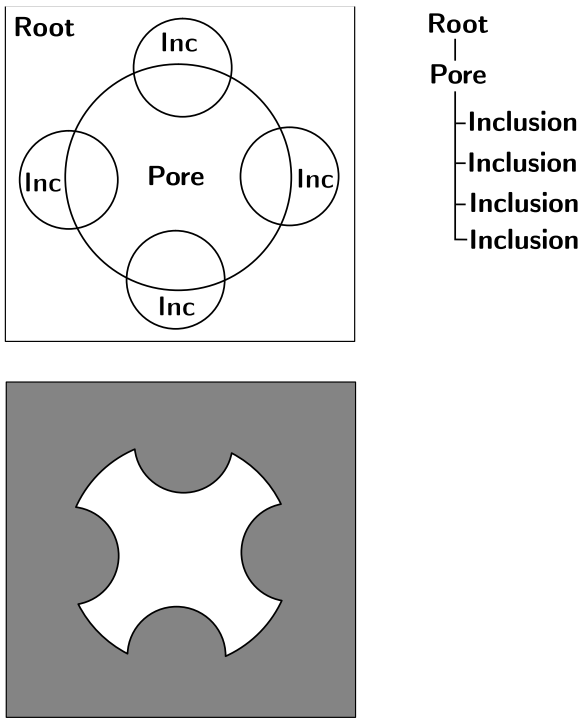


Figure 1.2: A complex looking pore (bottom) drawn from a set of discs and a series of Boolean NOT operations.

Constructive solid geometry is an efficient representation if the essential property of a geometry is the delimitation of a subspace. To test whether a point is in a geometry, all the sub-geometries in the Boolean tree are tested and the resulting Boolean expression evaluated. In the case of the setup shown as an example in Fig. 1.2, this expression is for point p:

$$p \cap (\text{Root} \cap \neg (\cap (\neg \text{Pore} \cap (\neg \text{Inclusion})))) \tag{1.1}$$

Although the API provides important flexibility for the formulation of any kind of problem geometry, the formulation of a problem requires the specification of the behaviour of the various features.

Behaviour laws

A `Behaviour` is an integro-differential expression linking two basis functions contained by an element. An element need not be a definite geometrical shape for the purpose of expressing behaviours, it only needs to provide a set of function, point pairs, and a method to compute the integral of a function.

A method of the object representing the behaviour returns a matrix which represents the linear interaction between the degrees of freedom described by two given basis functions. Because the expression and implementation of the behaviour law is not dependent on the type of function, it is very well suited for the implementation of XFEM.

The abstraction provided by the function class and the differential operators make it extremely easy to implement behaviour, even non-linear. The syntax was designed to resemble as much as possible the syntax of analytical expressions. As an example, the implementation of the Kelvin-Voigt and linear elasticity behaviour are shown on the Listing 1. This example shows how a time dependent and time-independent behaviours can be expressed in the same way.

```
Matrix KelvinVoight::apply(const Function & p_i, const Function
    & p_j, ...) const
{
        return vm->ieval(Gradient(p_i) * C * Gradient(p_j, true)
            , ...)
             + vm->ieval(GradientDot(p_i) * eta * Gradient(p_j,
                 true), ...)
             + vm->ieval(Gradient(p_i) * eta * GradientDot(p_j,
                 true), ...);
}

Matrix LinearElastic::apply(const Function & p_i, const Function
     & p_j, ...) const
{
        return vm->ieval(Gradient(p_i) * c * Gradient(p_j, true)
            , ...) ;
}
```

Listing 1: Implementations of Kelvin-Voigt and linear elastic behaviour.

[5]In practise, the behaviour laws are a recipe for the generation of elementary matrices. The elements are in this approach only a convenient way of associating basis functions to nodes. However, the elements provide the quadratures required for the computation of the discrete form of the problem.

Elements

The elements in AMIE are abstracted as collections of nodes with associated shape functions. This way, as there is no assumption on the number of degrees of freedom contained in each element it is easy to formulate problems with mixed element types, or to have enriched elements.

Elements also provide neighbourhood information. In many algorithms, the neighbourhood information is required. For example, when damage is computed, the fracture criterion calculated in an element is compared to the criteria computed in its neighbours. In post-processing, fields can be smoothed using the neighbourhood information.

Specialised elements exist also as definite geometries and parts of meshes. This is the case for the `DelaunayTriangle` and `DelaunayTetrahedron` elements, which are produced from the generation of unstructured meshes from the built-in mesher. The specialisation is useful as certain algorithms can be made more efficient if specific kind of geometries can be assumed.

1.3.2 Solver

The choice of solver is normally tuned to the specific application which is considered. However, AMIE was built with the assumption that problems would mostly be solved without external user input. The solver is set automatically by the programme as a compromise between pure performance and robustness, with a preference for robustness. The following solvers were implemented:

1. A Gauß-Seidell-successive over-relaxation (SOR) solver. This solver can only solve positive definite problems[6], and is rather inefficient. It is an iterative solver, so it allows the solving of large problems. In terms of convergence, it has a desirable property: it is absolutely convergent, meaning that at each step, the current solution is a better

[5]A quadrature is an approximation rule for the computation of an integral. In general, a function is evaluated at a series of points and the quadrature is a weighted sum of the results.

[6]A matrix $\mathbb{M}$ is positive definite if $z^*\mathbb{M}z > 0$ for all non-zero vectors z, where * denotes the conjugate transpose.

approximation of the actual solution than the one from the previous step.

2. A Cholesky decomposition/triangular solver. This solver is direct, which means that it obtains the solution in a finite number of steps, and suffers from no stability problems. However, being a direct solver it cannot be used over large problems because it consumes too much memory.

3. Conjugate Gradient (preconditioned). The conjugate gradient algorithm is iterative and fairly stable [44]. It converges in a fixed number of iterations to the solution. However, using preconditioning, a "good enough" solution can be found in much less steps. The preconditioner can accelerate the convergence considerably, however it may come at the cost of stability. The preconditioner used commonly is the inverse diagonal preconditioner[7] because it is absolutely stable. The conjugate gradient can only be used to solve positive definite problems.

4. Polak-Ribière iteration [45]. This is a modified conjugate gradient for non-linear problems.

5. Bi-Conjugate Gradient Stabilised [46]. This solver can solve non-symmetric problems. It is however less stable than the normal conjugate gradient.

The solvers have been implemented with a programming interface which fits well with the rest of the framework. They are not using specialised libraries for the linear algebra operations, but the performance is still adequate as the time-limiting operation is input-output (IO)-bound. Fig. 1.3 reports The megaflop rate of the matrix-vector multiplication in the node-parallel and single-processor cases. The sharp fall-off in performance is due to suboptimal cache usage. A sustained megaflop rate could be obtained with appropriate sub-domain decomposition. However, the fraction of the peak obtained for the systems benchmarked here — unstructured and sparse — is consistent with the best values reported in the literature. This is not surprising as the efficiency of the computation is entirely dominated by the bandwidth to the main memory and depends very little on the merits of the implementation.

[7]A preconditioner reduces the condition number of a matrix by multiplying it with a pseudo-inverse, which then reduces the number of iterations required to reach the solution.

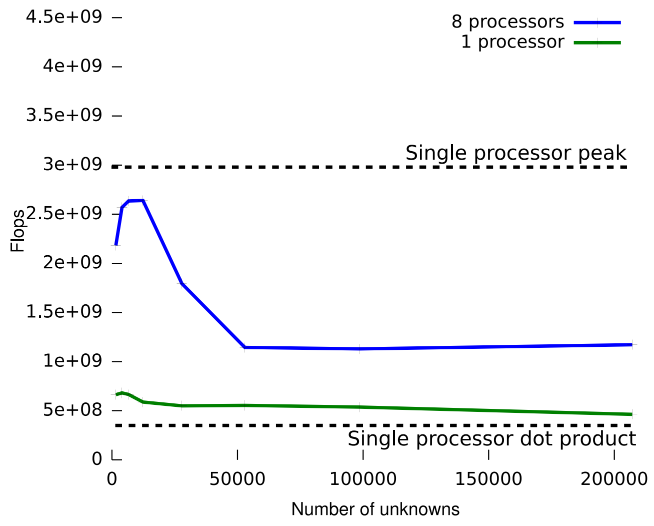


Figure 1.3: The megaflop rate of the matrix-vector multiplication is reported here in the node-parallel and single-processor cases. The fall-off in performance is due to problem size becoming larger than the processor cache. However, the fraction of the peak obtained for the systems benchmarked here — unstructured and sparse — is consistent with the best values reported in the literature.

The choice of solver is not performed by the user. AMIE provides low-level access to the solvers and the preconditioner as plug-ins. However in the main loop, the solver is chosen automatically from the problem type: linear or non-linear, symmetric or not. Problem type detection is done during the assembly of the problem matrix.

The performance of the solver is dependent on the availability of sparse matrix and vector algebra operations. A small library for such operations has been implemented. This library makes extensive use of operator overloading so the solvers can be implemented directly from pseudo-code as typically reported in literature. Further, this linear algebra library has been kept simple and well separated from the core to allow an eventual replacement using standard implementations of the portable, extensible toolkit for scientific computation (PETSc) or the linear algebra package (LAPACK).

XFEM Implementation

Modelling of the ASR at the microstructure level requires the use of numerical techniques which allow the representation of fine geometrical details within the microstructure. Further, it is desirable to be able to avoid re-meshing, because it is numerically costly, and field transfer between meshes can be the cause of numerical errors. A method which has the desirable properties of not requiring re-meshing and which yields good numerical approximations even for relatively coarse meshes is XFEM.

The behaviour of the numerical sample can be altered by modifying the space in which the solution to the Galerkin problem is sought. The method for deciding which, and how elements should be modified is called enrichment. It consists first in finding which elements are affected by a feature which changes the function space – for example, a crack which introduces a jump in displacements. Then the method adds to the nodes of the element shape functions which extend the solution space.

This method is closer to the Galerkin approach than the finite element restriction because such functions are typically global: an inclusion or a crack are defined in the global coordinate system, and their effect is also global. Compactness is retained by multiplying the enrichment function by compact shape functions, supported by the elements which are enriched.

The transformation of coordinates implied in this approach is greatly helped by the fact that it can be automatically expressed as a `Function` object in AMIE. In the examples in the following sections, the $\mathbf{x}$ is the global coordinates transformed into the local system of the element (ξ). The transformation is directly obtained from the linear Lagrangian shape functions of the elements, N_i, and the node coordinates $\mathbf{p}_i$:

$$\mathbf{x}(\xi) = \sum_i N_i(\xi)\mathbf{p}_i \tag{1.2}$$

Soft discontinuities are the enrichement type which was used for the simulation of gel pockets in the simulations. They are implemented by the introduction of a hat-like function, with the maximum located at the interface of two materials. This function is C^0 continuous, but C^1 discontinuous. Hence the term "soft" discontinuity. Such an enrichment is useful for the modelling of inclusions, or, in the case of ASR, gel pockets.

This type of enrichment simulates a perfect contact between two materials of distinct mechanical properties. The enrichment function ϕ used was introduced by Moës and colleagues [47].

$$\phi(\mathbf{x}) = 1 - |\mathbf{x} - \underset{\Gamma_{\text{inclusion}}}{\text{proj}}\ \mathbf{x}| \tag{1.3}$$

With $\Gamma_{\text{inclusion}}$ the inclusion boundary, proj the projection operator, and $\mathbf{x}$ coordinates in the global system.

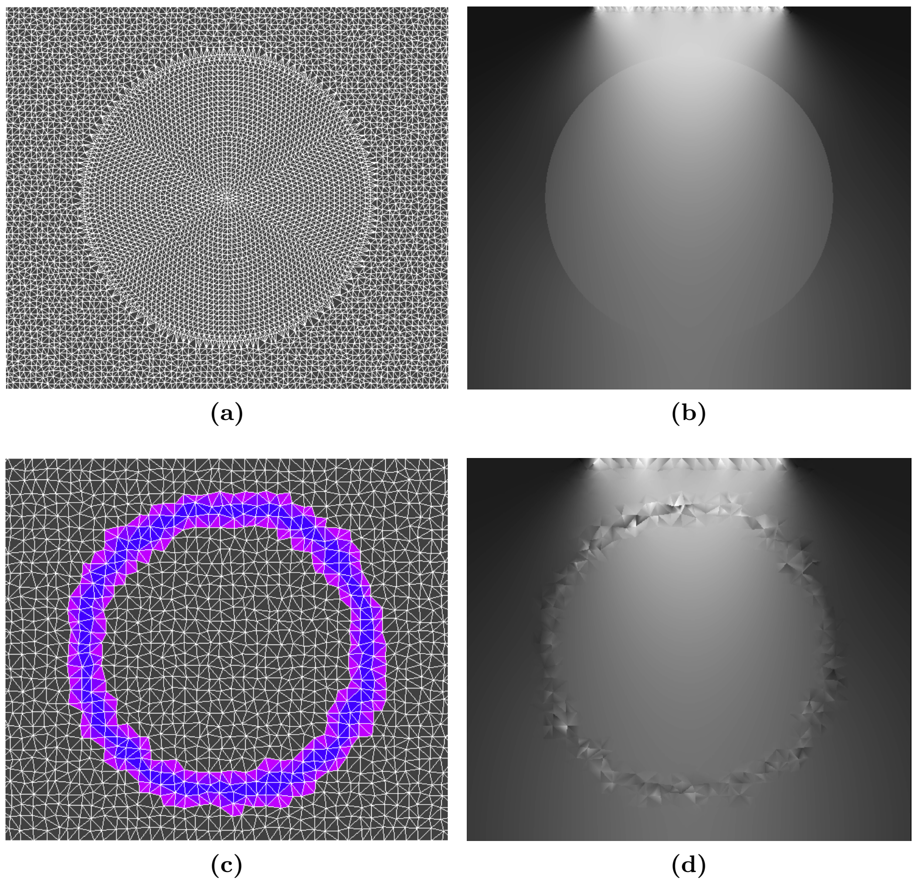


Figure 1.4: Comparison between conforming ((a) and (b)) and enriched ((c) and (d))approaches. The inclusion is has a Young's modulus four times that of the surrounding material. The radius-of-inclusion-to-width-of-the-sample is $\frac{3}{4}$. The stress field in the XFEM case is only apparently less smooth at the interface. This is due to the post-processing which does not involve the computation of a sub-mesh to use for display.

We designed a simple benchmark with a single inclusion in a rectangular sample. The inclusion is either modelled with a conforming mesh or with XFEM. The mesh is then refined until convergence of the measured values is obtained (Fig. 1.4). The goal of this benchmark was to verify whether the implementation of the soft discontinuity was correct, and check whether the stress field obtained could be used for the simulation of damage. If the XFEM inclusions had produced maximum stresses higher than the conforming inclusions, the mesh around them would need to be refined.

However, it was found that the XFEM inclusions yielded smoother results than the meshed inclusions (Tab. 1.1 and fig. 1.5), although this seems not to be the case due to limitations in the display code. This convinced us that they could be successfully used to model such small geometrical features as the gel pockets induced by ASR.

Table 1.1: Computed average and maximum values for various representations, conforming and enriched. The enriched mesh converges faster, and the maximum values are similar to that of an equivalent much finer conforming mesh. XFEM is therefore a suitable method for computing local damage effects coming from fine geometrical details. $\bar{\varepsilon}_{xx}$ is the average strain on the longitudinal axis.

Mesh type	# of elements	max $\sigma_{\text{von Mises}}$	$\bar{\varepsilon}_{xx}$
Classical	8646	27 144.2	4.2404×10^{-08}
Classical	35 912	34 269	$4.250\,55 \times 10^{-08}$
Enriched	8646	34 269	$4.250\,55 \times 10^{-08}$

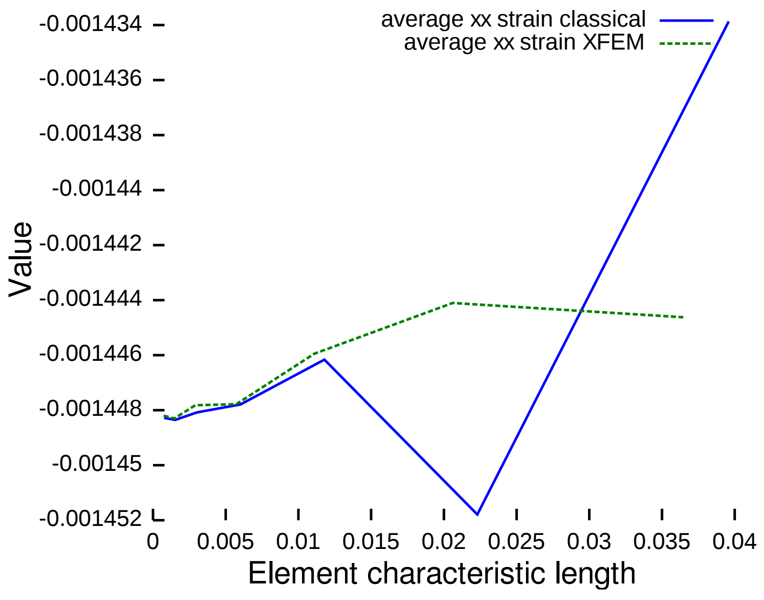
Further implementations of commonly used enrichment functions as well as the adaptive quadrature generation used for the integration can be found in appendix C.

The state machine of a simulation

The setup of a simulation is done by describing the geometry by adding `Features` to a problem. Each feature has a geometry and is attributed a `Behaviour`. The user then calls `FeatureTree::generateElements()`, and can apply boundary conditions as a function of the position of the element nodes. Once this is done, the user calls `FeatureTree::step()`.

The design of AMIE is synchronous. This means that all the elements of a simulation are undated at the same time, though in a definite order. To that effect, all the classes which have a state, for example material behaviour or XFEM describing cracks have a `step()` method. This method updates the state variable of the class, and calls the `step()` method of the classes depending on it.

- The `FeatureTree` `step()` method calls the solver and iterates on all the elements
- the `step()` method of the elements calls the method on the `Behaviours`
- The `Behaviours` update the `ElementStates` and if required call `step()` on internal classes of the elements



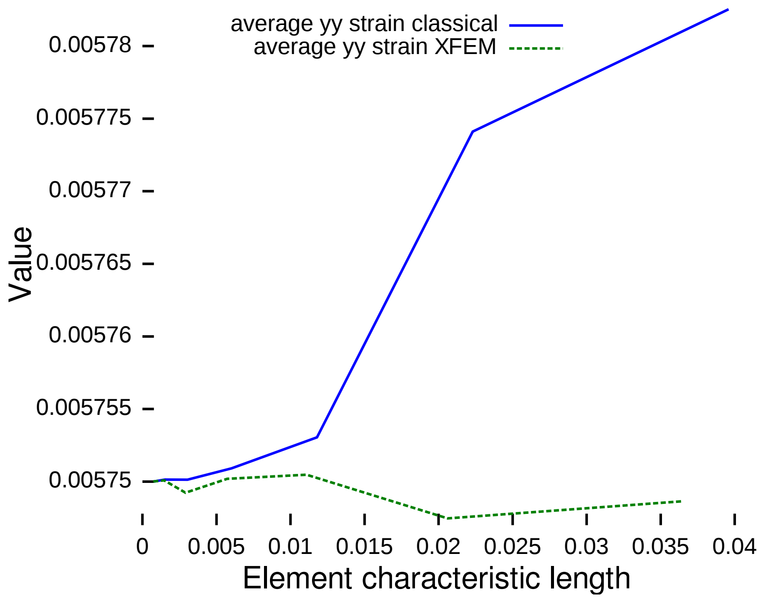


Figure 1.5: Convergence of computed average values as a function of the element characteristic length.

- `FeatureTree` calls the `step()` method on the `EnrichmentFeature`s.

Two state variables are then updated: the convergence of the solver, and whether any damage occurred. The user may use this information to either iterate until an equilibrium is found, or `stepBack()` the problem and change the boundary conditions.

Post-processing can be done using the methods provided by the `ElementState`. Those compute stresses, strains and displacement in the element using the Galerkin method for the reconstruction of the fields. This is less efficient than simply using the calculated displacements, but this small overhead means that the fields are computed correctly at any point in the element, even if it was enriched.

1.4 Mesher design

A crucial point in the design of an abstract simulation framework, designed to work with complex geometries is the availability of a mesher which can work without supervision. The mesher should be able to produce a mesh only from the geometry information. Further, the availability of an integrated mesher is necessary for the efficient implementation of the XFEM.

The meshers integrated with AMIE are Delaunay tesselators using an internal tree-like structure allowing point insertion in $\mathcal{O}(\log(n))$. The same structure, derived from geometrical considerations, makes it possible to also find triangles or tetrahedrons overlapping a [8]given geometry in $\mathcal{O}(m\log(n))$ with m the number of elements covered by the geometry. This is useful in:

1. localised refinement,
2. mesh-geometry interaction finding,
3. geometry-geometry interaction finding,
4. sub-triangulation for non-polynomial function integration.

1.4.1 Finding geometrical interactions

Computational geometry has applications in fields as diverse as ray tracing [48], games [49] and physics engines [50]. Efficient algorithms for the detection

[8]Big-O notation means that the time to perform an operation is proportional to the content of the expression in the brackets. For example $\mathcal{O}(n^2)$ mean the time taken is the square of n.

and localisation of various geometrical interactions have been published. The implementation of these algorithms depends on the availability of the data-structures used in the programme.

Representation

Enrichment features are well represented as deriving from general geometries, and providing a means to enrich the elements they affect. It is either possible to attach objects representing physical laws to them, or to consider them purely geometrical objects depending on which is more efficient in each circumstance.

The dependency of enrichments and mesh is the following: the element set is responsible for finding the subset of conflicting elements. If the elements are stored in an efficient data structure, the element set can find the interactions much more efficiently.

This shows the need for a mesh database more than for a mesher. However a lot of the code in the mesher is useful outside as well, and the core of the mesher can be structured as a database.

Computational geometry provides the information on the relative position of a point and a given geometrical entity. It can also provide the location of intersections. Both informations are needed to decide whether an element is a target for enrichment or not. Although the check whether a given element *is* a target for enrichment is computationally inexpensive, it is much more difficult to have an exhaustive list of *all* the targets. Indeed, the information is attached to a geometrical entity, and contains no topological information. In a FEM program, the mesh contains such information in the form of a connectivity table. Although it is sufficient to recreate a table of neighbours it is inefficient to do so: this structure is ill-suited for fast searches.

Special case for the elements

The elements are usually also geometrical entities, and so implement the interactions. This is necessary for enrichment, but higher order elements are not convex polygons, which induces some computational overhead. If the mesh is well-adapted to the geometry of the problem and does not rely too much on the additional degrees of freedom provided by the higher-order elements to ensure an acceptable approximation of the boundaries, the elements can generally be treated as their low-order counterparts, making the process of finding the enriched nodes much faster. The process of finding the enriched elements is dominated by the efficiency of the `in()` method.

This method would be most efficient if elements were circles or spheres, with only three multiplications and a comparison required.

This is nearly the case when the mesh has been generated by Delaunay tesselation. Such a tesselation is defined by the following property: *No four respectively five points are within any of the meshes triangles' circumcircles respectively tetrahedrons' circumspheres.* Other meshing schemes, which are not based on circles or spheres interactions can still be stored in a efficient tree-like structure, but the exact implementation of the `in` condition is going to be either less flexible in the case of a purely regular rectangular mesh, or more costly.

1.4.2 Description of the meshing scheme used

Two kinds of meshing schemes dominate in practise, the advancing front and the n-Tree family. They both have advantages and disadvantages when considered strictly from the meshing point of view, but for our purpose, the advancing front family is not well suited.

Typically the input consists of only the analytical description of the geometry, and it is up to the mesher to do the sampling. Alternatively, the sampling is given, and the mesher then outputs the elements corresponding to the given sampling points, constrained by the boundaries of the geometrical entities to mesh.

Trees are an efficient mean of organising and retrieving data [51]. In particular, for space-dividing schemes, a parent-children relationship ordered by the spatial position allows the building of such trees on the fly as the meshing goes. A very simple example of such meshing is a quad-tree (Fig. 1.6).

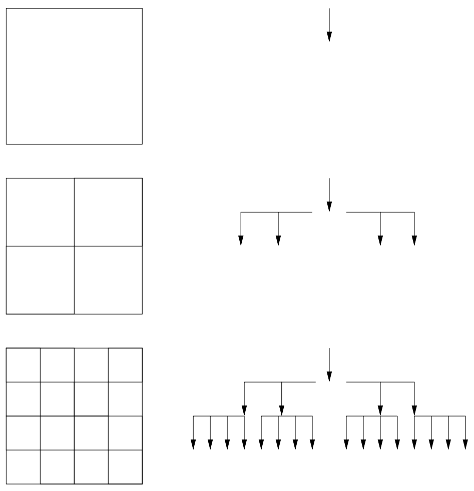


Figure 1.6: Illustration of a simple quad-tree. The children are all contained in their parents. This makes searching very fast, as each test divides by four the amount of elements yet to check.

But a tree is also a possible representation of the current state of a Delaunay type triangulation. Points are pre-generated and then fed randomly into the triangulation algorithm. Each new insertion determines the elimination of a set of triangles, and the creation of new ones [52]. The new triangles are the children of the old with which they share the same conflict condition. A schematic illustration of the process is found in Fig. 1.7. The algorithm works thus:

1. if we already checked this element, return,
2. if this element is not conflicting with the point, return,
3. add self to return,
4. attempt insertion in all children, insert result,
5. attempt insertion in all neighbours, insert result,
6. return.

Once a list of triangles is obtained, they are deleted and replaced with new triangles defined by the boundary of the set of deleted triangles, and the inserted point. The combination of the conflict-checking algorithm and the point insertion scheme generates a set of triangles, arranged in a tree whose leaves form the final mesh.

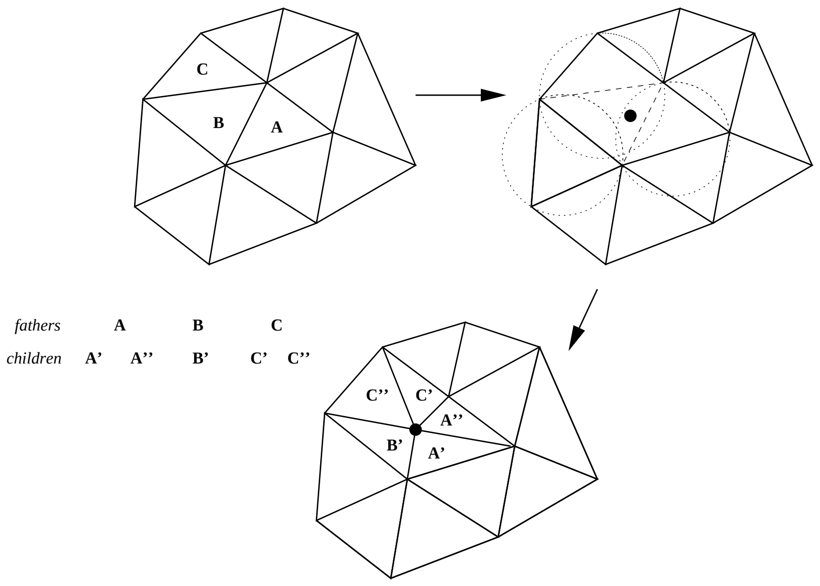


Figure 1.7: Schematic representation of the insertion of a point in a delaunay triangulation.

Mesher efficiency

The mesher we coded is fairly efficient. We compared its performance to gmsh [53] which is widely used as a research tool, and got the following results:

Table 1.2: Time taken for a mesh with ≈ 50 000 points which finally yields 200000 degrees of freedom

gmsh	AMIE
5 s	7.4 s

Gmsh is 30% faster, but the order of the algorithms is the same. We conclude that our mesher is *good enough*, though it could certainly be optimised. The disadvantage of lower speed is offset by having full control over the API and the additional usage we get from the mesher, using its readily available data structures. Further, the meshing step is in fact a relatively low-time-consuming one, as highlighted in Fig. 1.8.

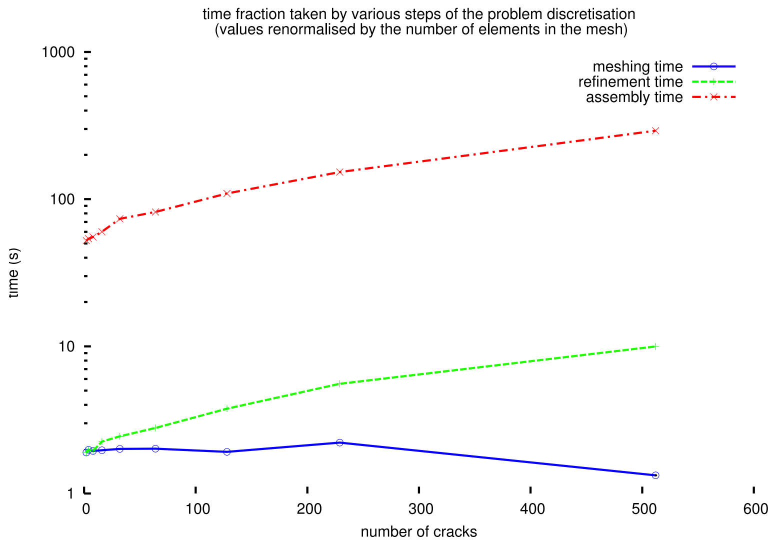


Figure 1.8: Time taken by different operations in a simulation. Localised refinement and matrix assembly are much costlier operation than the meshing itself. The renormalisation highlights the asymptotic behaviour of the various algorithms involved.

1.4.3 Comparison with a mesh database

A mesh database, such as algorithm oriented mesh database (AOMD) [27], would provide the same accessors and iterators as the mesher, thus would be equivalent from the point of view of efficiency of the mesh-geometry interaction detection step. However, the database would need to be rebuilt each time the mesh is rebuilt, leading to unnecessary overhead. The algorithmic advantages are nonetheless the same as for static meshes. But the overall algorithmic efficiency can never be better than $\mathcal{O}(n)$, where n is the number of elements, as each element is to be entered at least once.

Once a mesh has been generated, it is fully defined by (1) the coordinates of the nodes, (2) the list of elements defined by a list of nodes. In a typical FEM code, only this information is exported and loaded in the assembler-solver. This is the *external representation* of the mesh. However, as we have just seen, much more information is used in the mesher. This information is typically discarded in the toolbox approach case, as it is useless: separate programs provide separate services and communicate through data files containing only the minimum necessary. However, one can build all sorts of element-finding routines on the model of the `conflicts(Point *)`. This only requires the programmer to provide a function checking for geometrical conflict. Enrichment schemes are determined on a geometrical basis. Elements in and around enrichment features are selected, and additional degrees of freedom

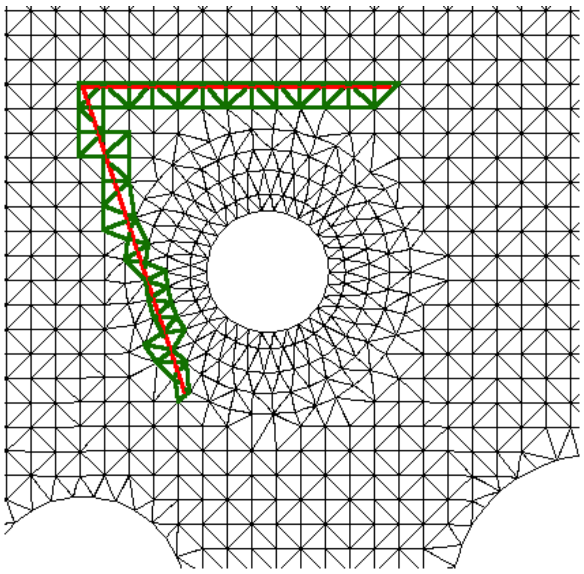


Figure 1.9: Detecting a geometry in a non-trivial case. Detection of a crack interacting with the elements, in bold, the elements intersecting the crack path.

are added to their nodes, based on the nature of the feature. This typically means that a given feature should be located in terms of elements. This is easy in the case of a so-called *structured mesh* where the spatial position of nodes determines their numbering. It is however much harder in the case of unstructured meshes, as those typically produced by the meshing scheme described above, used in practise. With only a list of elements, the only way to find those within a given geometrical limit is to run through the list, and check every element. This is dramatically inefficient. The slightly better case involves having also a list of neighbours for each element. Then, one needs only to find a single affected element and proceed through the neighbours. This is faster, but *is still asymptotically the same order of computation* because the first element still needs to be found, and this is $\mathcal{O}(n)$ for randomly listed elements. Indeed, on average one will still have to check for half the elements before finding one that matches the condition and only the constant part of the iteration is reduced.

Finally if the elements are arranged in a tree, the order of the computation is dramatically reduced from $\mathcal{O}(fn)$ where f is the number of features and n the order of elements, to $\mathcal{O}(f\log(n))$. In Fig. 1.9 the elements crossed by a segmented line have been marked. This is exactly the operation which needs to be optimised for XFEM applications.

Fracture mechanics and *a posteriori* error applications

Modern fracture mechanics applications and schemes require a range of geometrical routines which are very easily provided using the architecture suggested. It is necessary to evaluate domain integrals for the computation of stress intensity factors. This requires finding elements within a ball of given radius, or, alternatively, elements cut by a sphere. This is trivially accomplished here. But even more interesting is that when using special shape functions, the integration step involves generating a sub-mesh on the elements, and summing the sub-integrals. This is typically more precise than simply using a very large amount of Gauß points. In addition, the idea of "fixed area enrichment" [54] rests on enriching sets of elements within a given surface area, a ball centred on the crack tip. Using the above general geometry description, the implementation and integration of such a scheme becomes very easy. *A posteriori* error estimations such as ZZ-type error estimates typically require knowledge about nodes' neighbouring elements, in order to construct the enhanced stress and strain fields on the computational mesh. Other error estimators [55] also require identifying nodes within a ball centred on the computational point where the smoothed field or derivative is desired. Again, the mesher integration is the easy answer to the program architecture challenge. Finally, one can cite meshless methods based on moving least square (MLS) approximations which also typically require finding neighbours, information which is directly stored in the mesher.

The possibility of error-based remeshing [55] is also very powerful. For time-dependent calculations, a once well-adapted mesh might becomes inadequate and yields faulty results. This leads on one hand to a necessary remeshing, and on the other to the recomputation of the mesh-geometry interactions. This cannot be done efficiently by using a mesh database, as it would have to be rebuilt with the new, modified mesh, an operation which is needlessly costly.

The API is the collection of functions and data-structures available for the programmer's use. Their richness and design make the difference between some extensibility or no extensibility [37]. The API of the integrated mesher must be able to provide the required services. However, for architectural reasons, it needs only know about *geometry*, and not specifically about elements or physics. The most important service provided is getting the list of elements affected by a geometrical feature. This is all that is necessary to get computationally efficient enrichment strategies when they depend on geometry.

The availability of a mesher also allows the implementation of fast global damage computation algorithm as presented in the Chap. 2, by providing a fast way to find neighbourhoods of elements which must be compared when finding the localisation of the damage.

CHAPTER 2

Damage Model Implementation

Parts of this chapter have been submitted for publication in the Journal of the Mechanics and Physics of Solids.

2.1 Introduction

Microstructural simulation of alkali-silica reaction (ASR) requires the availability of a numerical tool capable of simulating dense cracks patterns efficiently. Such a method should also be capable of computing the crack patterns from the final applied load at each step of the simulation. The treatment of damage which allows such calculations is presented in this chapter, most of which has been submitted for publication.

Finding the damage state of a material sample from a history of boundary conditions is a difficult problem, theoretically [56] and numerically [57]. It is theoretically difficult, because the problem can be ill-posed [58], and the existence and uniqueness of solutions is not necessarily demonstrated [59]. It is numerically difficult, because dramatic changes in the state of the material need to be tracked [60].

The algorithm presented here was developed with the specific goal of computing the degradation state of concrete by various mechanisms. Degradation mechanisms in concrete are typically characterised by the induction of internal loads which are geometry-dependent. Examples of these are reaction fronts behind which the concrete expands. Further, those degradations of chemical origin change the nature of the cement paste and its fracture behaviour. Also, concrete is a composite material, with aggregates typically four times stiffer than the surrounding paste. When those are taken into account, problems are further complicated by the presence of aggregates which might themselves fail.

As fracture can be induced by local changes of mechanical properties of the material, computing a critical load multiplier with the goal of following the equilibrium path of the material is not generally possible: there might be no equilibrium state, for example the sample may fail completely. Also, as the load can be induced by a change in the internal geometry of the specimen, continuum damage models do not formally apply: the computational domain might change discontinuously for example. In certain phenomena chemical changes occur within the domain, causing local phase changes. One such case is the alkali-silica reaction (ASR) in concrete.

This problem is further complicated by the fact that a damage model is typically scale-dependent [61]. We propose a method which yields reasonable solutions and tends to the energy minimum solution as the discretisation scheme is refined. This method is validated in simple and complex cases by numerical examples compared to physical experiments and the literature. This algorithm resembles the sequentially linear analysis method [62], but differs in that it does not require the computation of load multipliers.

2.2 Damage Simulation Algorithm

Non-local damage models such as those presented by Jirásek and coworkers overcome the mesh-dependence of local expressions by introducing regularised fields [63]. The regularised fields are obtained by introducing a global weighting function which averages the values on a fixed-size domain. The averaged values are then used to compute a failure criterion, and damage is distributed using the same weight function on the elements contained in the neighbourhood [64]. Such an approach has several advantages: it is mesh-independent (provided the localisation zone is larger than the elements), also in term of released energy, and the critical length can be set to physically represent the observed fracture process zone [65]. The size of this process zone, however, is material dependent, and in general not precisely known. But there are also other drawbacks to such approaches. The regularisation of the fields assumes a homogeneous material, and thus is not well-suited to the simulation of explicit microstructures [57]. Another limitation is that there is no causality constraint[1] on the attribution of damage: damage states are considered, but not damage histories during the loading step. Thus, boundary conditions have to be compatible with the computed damage state. In arc-length solvers, $\{\text{damage}, \text{load}\}$ couples are solved for. However, if the loading steps are imposed, or it is impractical to iterate on the load, this can be a severe limitation. This approach is not well-suited to problems where the geometry of the boundary conditions is externally imposed: the regularisation then becomes difficult since the averaging domain required for non-local models is time-dependent. A practical example of such a problem is microstructural ASR simulation [41] where aggregates transform into expansive gel.

To address some of these difficulties in simulating failure of materials, a number of techniques co-exist in the literature. Perhaps one of the most general and promising approaches was proposed by Francfort and Marigo [60] who revisited brittle fracture as an energy minimisation problem. This generalizes Griffith theory by permitting (i) to handle brittle crack initiation; (ii) to track instantaneous brittle failure [66]. Numerical experiments using this theory were presented in [67], extended to cohesive cracks in [68], and reviewed in [69]. This novel approach is particularly appealing as it allows to simulate crack nucleation and to obtain all possible crack paths as a direct outcome of the simulation, while ensuring energy minimisation. This is critical when multiple cracks initiate and propagate, which is the case in microstructural concrete simulation. A closely related method which relies on the global Clausius–Planck inequality in the sense of Coleman's method and configuration forces [70] was proposed in 2009 by [71]. This framework relies on maximising the local energy dissipation along the crack fronts, yet, as in [60], the minimisation process is handled globally. Similarly, Dumstorff and Meschke [72] exploit the extended finite element (see below for a more detailed description) to simulate multiple (cohesive) crack propagation through a global energy minimisation, as proposed in [73].

The algorithm proposed here belongs to the same class of "global energy minimisers" but differs in that failure/degradation is not modelled explicitly using strong discontinuities (see below for details), but by a damage law. Indeed, this algorithm was developed with the specific goal of computing the degradation of concrete microstructure, where high crack densities significantly complicate the book-keeping of fracture patterns, thus justifying damage-based approaches.

The proposed modelling framework was designed to be most efficient for simulating failure mechanisms in complex microstructures especially

[1] In non-local damage models, damage is distributed in the material in a single step, using an *a priori* selected weight function. It is thus not possible to capture the effect of progressively softening material domains actually leading to its final damage distribution.

when these are characterised by the induction of internal loads which are functions of the evolving geometry of the microstructure. An example of such microstructurally-driven degradation mechanisms is the alkali-silica reaction (ASR), where silica dissolves on the surface of the aggregate particles, leading to the formation of expansive gel pockets and the consequential nucleation of micro-cracks. This dependence on time and loading of the microstructure complicates the simulation process, and it is a non-trivial task to ensure that the computed solution corresponds to maximum energy dissipation.

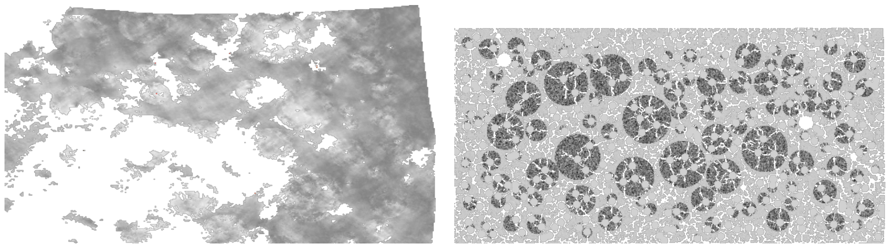

(a) Local damage attribution **(b)** Non-local damage attribution

Figure 2.1: Damage patterns for local (a) and non-local (b) damage attribution algorithms. Non-local damage attribution favours the appearance of distinct crack patterns, whereas local attribution causes un-physical destruction of the sample. In both cases the inclusions are for times as stiff as the surrounding matrix and expansive pockets have been placed randomly in the aggregates.

Fig 2.1 compares two simulations in which damage is induced by internal stresses localised in pockets located within inclusions. The setup is composite, with a matrix, inclusions, and inclusions within the inclusions, all placed randomly. Fig. 2.1b shows the result when the attribution of damage is both determined locally and applied locally. Fig. 2.1a is the same setup but with attribution determined on a global basis and but still applied locally.

This example suggests that the criteria and irreversibilities should be applied locally, so that composite problems can be considered, but that the choice of the loci for the irreversibility increment be computed globally. It is necessary to consider the global problem to find where damage occurs, because failure to do so leads to unphysical results such as the destruction of large sections of the domain, however local increments are then sufficient to find the damage state of the sample. A suitable renormalisation of the criteria allows comparison of subdomains with different modes of failure, and an iterative procedure is defined which minimises the energy dissipated in the material transformation. Scale should be handled at the material law level: the damage mechanism should explicitly depend on the element size. Using this approach, the localisation of the fracture appears spontaneously from the expression of the damage process in the material.

2.2.1 Energy Relaxation Mechanisms and Elastic Limit of Materials

Classical yield criteria such as Mohr-Coulomb or von Mises have been used to predict the limit load at which a material starts failing [74]. In general one can establish a criterion describing a yield surface in the state space of the material such that deformations beyond that surface are partially irreversible [75]. This criterion is a measure of the energy necessary to initiate a material transformation. It can be expressed as a test on a scalar field in the space of the element state.

When a material reaches its yield surface, this surface evolves so that the material is never beyond it. For a given loading condition, computing the elastic deformation of a domain can produce a result where domains of the material are beyond the yield surface. This indicates that part of the material would undergo irreversible transformation under such loading conditions. However the only certain relation between the domain beyond the surface when elastically deformed and the domain of the material effectively transformed by the loading is that they must have a common subdomain.

The energy used for the irreversible transformation is provided by free energy available to the material, such as the elastic energy from the deformation under load or heat sources. At the material level, not all elastic energy can be used to transform the material. The part/type of the elastic energy available is governed by the choice of the yield/failure criterion. In von Mises plasticity for instance, only deviatoric deformations can cause yielding. The yield criterion links the state of the material to the occurrence of irreversible transformation. The criterion value, usually obtained by a trial linear elastic analysis grows *monotonously* with the stress, and thus with strain and energy.

In the domain which is beyond the yield surface in the trial elastic deformed state, the material has been affected and some damage/plasticity/cracking must have occurred. From the above considerations, for a given loading condition, the subdomain in the material in which the damage/-plasticity/cracking criterion is most exceeded has undergone irreversible transformation. The identification of this subdomain is key to the proposed algorithm described below.

The formal proof of the convergence of this algorithm is not done. We present the physical basis for the formulation and a series of numerical experiments showing that the proposed algorithm has the expected properties. Not all experiments are presented in this chapter: the rest can be found in the App. E.

2.2.2 Formulation of the Degradation Process

Given its history and state, a material can only get to certain admissible states. For example, damage is usually irreversible, cracks do not heal, plastic deformation stays plastic. If one considers a finite set of states for the material, representing for example the presence of an increasing finite number of micro-cracks, the number of possible transformations a domain can undergo is finite. The possible material states are then defined from the current state and a suitable update rule must be defined for the material behaviour (including crack initiation, etc.).

The nature of irreversibility is not scale independent in composites (see for example Tjrnlund and colleagues [76]). This means that the damage evolution, and thus the admissible states of the material are not independent of the volume considered. Indeed, a material can be ductile at the macro-scale and brittle at the micro-scale. It is assumed that the local increments in irreversibility computed from the material model are such that the material considered is in an admissible state at all considered scales. This implies that the possible increments must be determined on a per-element basis, as they depend on the size of each element.

The local energy states of the material can be ordered such that two consecutive states differ by a predefined energy increment. This increment in energy cost, associated to each possible transformation, allows shifting between two successive material states. From a given state, physically, one can only move to a state which is of lower elastic energy (i.e. larger dissipated energy). When minimal-energy solutions are sought, from a given state, for a given energy increment used in transforming the material, only the states accessible through continuously decreasing energy are thermodynamically possible [77].

This defines an energy landscape for the possible transformations at the global level. Because the yield criterion is monotonously increasing with energy, the path defined by maximum energy gradient descent is close to the path of maximum yield criterion gradient descent. From the definition of the yield criterion, at the yield surface, those two paths are identical. If the irreversibility is described by a single variable (such as for isotropically damaging materials), the path is also unique. Importantly, this approach is valid also if the material domain is composite: the set of states is defined locally. This allows simulating composite microstructures.

2.2.3 Energy Minimisation

Existence of a Solution

The final state of the material under a given load is the result of the thermodynamically admissible set of local state changes which minimises the energy stored in irreversible deformations. If irreversibility is locally incremented where there is the most available energy for transformation, a small increment in irreversibility at that location will cause the least relative variation in locally available energy, but the maximum variation in terms of global energy.

It is necessary to use sufficiently small increments in the irreversibility, because the alternative is the minimisation of this energy using explicit energy functionals, which might not be experimentally measurable, or computationally expensive to calculate [60].

For practical purposes, however, it suffices that the locus of the series of domains where irreversibility was incremented exists uniquely with reducing element size. The solution to the problem consists in the values of irreversibility state at all points of the domain. Thes values are such that equilibrium is reached and the minimum amount of irreversibility is obtained.

In the case of a perfectly brittle material with only two admissible states, the damage increment admits a single value, 1, and the volume of the damaged domain should tend to 0 as the elementary volumes sizes tend to 0. As the damaged volume tends to 0, the energy needed to form it tends also to zero, because the criterion is formulated for volumes. Therefore, we must impose the size of the smallest possible convex volume which behaves as *effectively* perfectly brittle. This is similar to the so-called smeared-crack approach: the crack formation energy is assumed to be dissipated over the whole element. This is consistent with our approach of formulating the material law by explicitely taking into account the scale of the domains considered.

Quasistatic Case

As the volume of material affected by a step of the process is imposed, the solution can be described either as a geometrical locus with defined boundaries, or by the field of irreversibilities at equilibrium. In the quasistatic case, dynamic effects such as inertia are neglected.

Consider a loading step, over a time slice where the load is monotonously growing. The initial load is such that the deformations are elastic. The material is then tested elastically with the final load and locally reaches its irreversibility limit. Thus, at an intermediate load, the irreversibility limit is reached in an arbitrarily small domain. An irreversible transformation of the material properties in this domain is possible, and dissipates an energy equal to the difference in the elastic energies stored between the modified domain and the original domain. This transformation need not be at equilibrium: the energetic cost of the transformation need not be equal to the elastic energy variation, it must only be inferior. A new elastic solution can be obtained with the altered global domain and the same load.

At equilibrium the elastic energy variation in the sample must be equal to the energy needed to transform the material.

$$\Delta E_{\text{elastic}} = \Delta E_{\text{irreversible transformation}} \tag{2.1}$$

This is equivalent to the irreversibility criterion not being met anywhere in the domain, and $\Delta E_{\text{elastic}}$ is minimum.

At any step of the process, $\Delta E_{\text{irreversible transformation}}$ is the sum of all energies from the successive irreversibilities introduced in the material. As the variation in elastic energy is automatically found from the load and the current material state, if an equilibrium is found, the error in the energy release from the irreversible transformations is at most equal to one step of relaxation. Thus it can be made to be arbitrarily small.

The irreversibility occurs in individual elements, thus the precision of the final solution depends on the mesh fineness: a coarse mesh will lead to a low resolution of the locus of damage.

The stored elastic energy during the iteration can only go down, thus the algorithm always converges if the problem stays well-posed as the material is modified. It is not trivial that it converges to a unique solution, independent of the successive choice of domains because although the elastic energy always diminishes everywhere in the domain, the free energy can locally grow as the stresses are redistributed. The examples show that the algorithm does indeed converge.

2.2.4 Algorithm Formulation

The fracture criterion $\mathcal{C}$ is usually computed from a function of the element state $\mathbf{s}$ and compared to a critical value, c:

$$\begin{cases} 1 & \text{if } \mathcal{C}(\mathbf{s}) > c \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

This function is smooth and monotonously increasing with $\mathbf{s}$. To use it in the algorithm, it is renormalised $\mathbb{R} \to [0, 1]$:

$$\begin{cases} 1 - \left|\frac{c}{\mathcal{C}(\mathbf{s})}\right| & \text{if } \mathcal{C}(\mathbf{s}) > c \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

This yields a smooth function ranging from zero to one, monotonously increasing with the distance from the fracture surface defined by the criterion, when the deformation is elastic. For example, a very simple criterion is a strain limit. If some equivalent strain measure $\bar{\epsilon}$ is larger than a threshold ϵ_c, the material should fail. So the modified criterion becomes $1 - \frac{\epsilon_c}{\bar{\epsilon}}$.

The update of the element can be done using a non-local damage model, in which case all elements in a ball centred on the element with the maximum value will be updated. Thus, the efficient implementation of this algorithm requires working in a numerical framework which provides facilities for fast access to elements within a geometric locus. Such an environment is typically a constraint on the architecture of the framework [41, 78]. The algorithm is given in Figs.2.2 and 2.3 and reads:

1. Compute the elastic deformation from the load.
2. For each element, compute the criterion using Eq. 2.3 as a real normalised value, for example, if the fracture criterion is a maximum strain ϵ_c, compute:

$$\nu = 1 - \left|\frac{\epsilon_c}{\bar{\epsilon}}\right| \tag{2.4}$$

3. For each element, define a neighbourhood in which to compare the values of the criterion.
4. If an element has the highest value in its neighbourhood, update the behaviour of this element. If the element is plastic, add some plastic strain; if it is perfectly brittle, remove the element; if a damage law is employed, increment the damage in this element. *Mark all other elements in the neighbourhood so they cannot be damaged at this iteration.*
5. Iterate from the new material state.

In general, it is not always desirable to sort all elements, but rather treat element neighbourhoods. This might be a better representation of physics, if the transfer of information in the medium cannot be assumed to be instantaneous, or if there is a characteristic distance beyond which

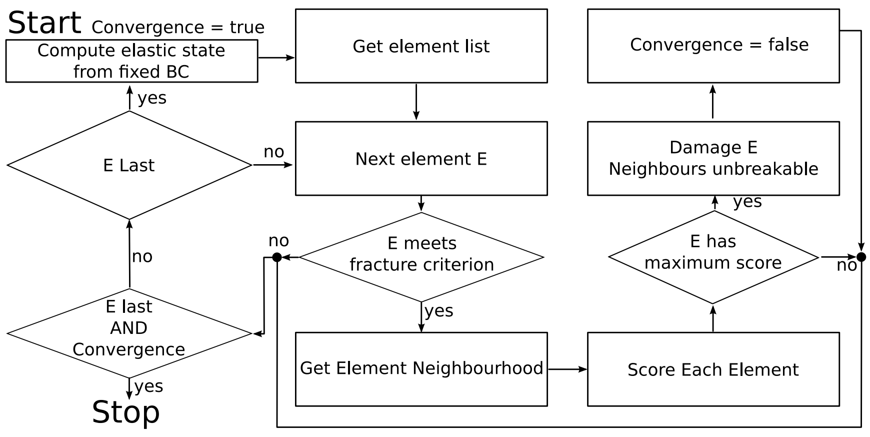


Figure 2.2: Damage attribution algorithm description.

perturbations can be assumed to have a negligible effect. This is a reasonable assumption when damage occurs throughout the material: small, local damage increments will not affect the solution if the material considered is geometrically complex.

It is necessary to ensure a single element is damaged per neighbourhood or is a centre for damage attribution: this ensures that the damage allocation is not more local than specified. Locality should be understood here by "distance between two damaged elements." In this sense, if more than one element were damaged per neighbourhood, the damage would be more "local", and damage could coalesce in a single band of elements within an iteration.

Composites under load have very complex stress patterns. The stress patterns are locally influenced by the aggregates and pores. This means that the influence of the weakening of an element is negligible within a radius approximately equal to the median radius of the inclusions. Such a setup is highly favourable for the damage attribution algorithm.

Because of floating point precision in microprocessors, there will almost always be a single element or none which should be damaged, according to the algorithm. This might not be physical: in some ideal symmetrical setups, multiple cracks should initiate and grow simultaneously. Since it cannot be expected that discretisation be perfect, nor that no numerical errors are introduced in the calculations, a tolerance needs to be introduced in the algorithm: damage is incremented in all elements with criterion value within δc of the maximum value in the considered neighbourhood.

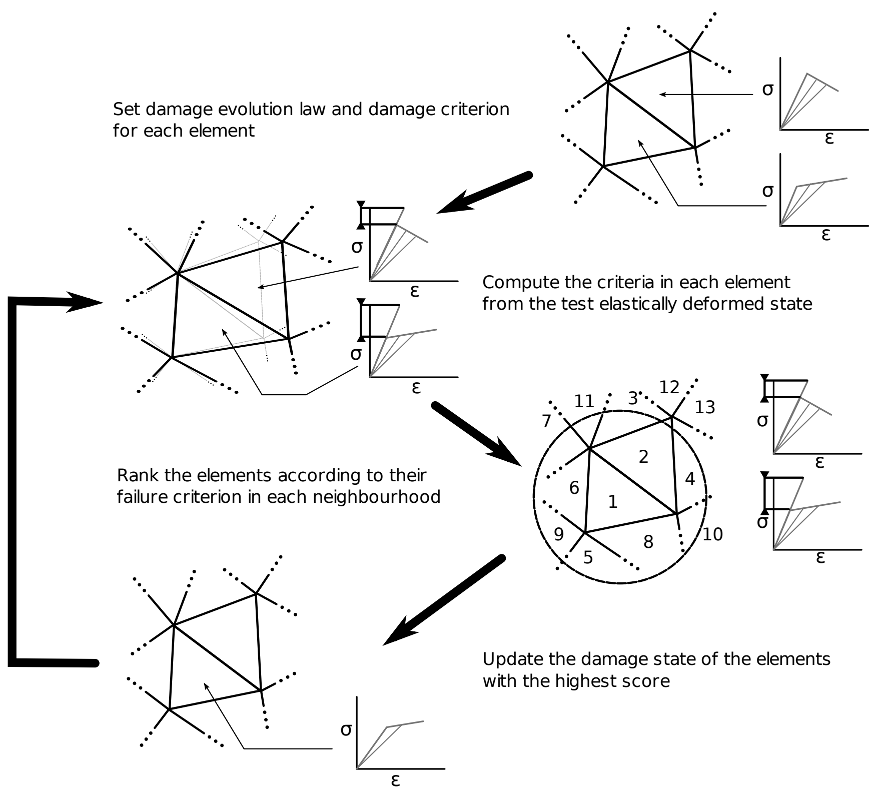

Set damage evolution law and damage criterion for each element
σ
ε
σ
ε
Compute the criteria in each element from the test elastically deformed state
σ
ε
σ
ε
Rank the elements according to their failure criterion in each neighbourhood
1
2
3
4
5
6
7
8
9
10
11
12
13
σ
ε
σ
ε
Update the damage state of the elements with the highest score
σ
ε

Figure 2.3: Schematic description of the algorithm for a hypotheticl particular case.

2.3 Proof-of-Concept Example

2.3.1 Setup

The algorithm performs very efficiently if the damage can be applied simultaneously in many places. This is the case for a complex composite microstructure, because the information of the deformation induced by the damage is quickly lost as the stresses are dissipated through aggregates and pores. However, to test the algorithm, we have conducted numerical experiments on simple samples, treating the damage attribution globally. This represents a worst-case scenario for this algorithm: it is designed to work efficiently when damage happens concurrently in many places at once, so equilibrium is reached within a small amount of steps. Here, damage can only occur in one or two elements at once, and eventually coalesces to a large domain, requiring many steps.

The algorithm converges experimentally as $\mathcal{O}(n^{1.72})$ where n is the number of unknowns. This convergence was obtained for a fully global iteration, where all elements are considered when choosing the locus of the damage increment. The time taken to find which element to damage is small compared to the time taken to solve the linear system of equations, thus the performance in terms of computing time of the algorithm depends on the choice of solver.

However, the dominant factor in terms of time is the total number of damage steps to perform. The time taken is proportional to the number of elements damaged times the number of damage steps in each element. However this can be considerably reduced if more elements are damaged at each step, for example by reducing the neighbourhood size: a more local iteration, suitable in the case of a complex composite yields considerably faster convergence. Further acceleration is obtained if the setup is such that damage occurs simultaneously in many elements. In the case of a sample subject to ASR with 200 000 unknowns, the updated damage state can be computed in less than 150 iterations per load step, instead of tens of thousands for the largest simple examples of this section with less than 40 000 unknowns.

The same numerical experiments were run with different mesh refinements and recorded the energy release with the algorithm advance. The sample was discretised with our in-house Delaunay mesher at four different mesh densities [41]. The goal of these experiments is to observe the effect of the choice of fracture criterion on the damage pattern, for materials with the same set of states.

The experiment setup is as Fig. 2.4. This setup, with additionnal initial cracks is usually used as a benchmark for partition of unity (PU) crack propagation. It is interesting because of the inbuilt symmetry and mixed-mode failure it induces. Also, this experiment has a very high phase contrast between the phases (pores and matrix).

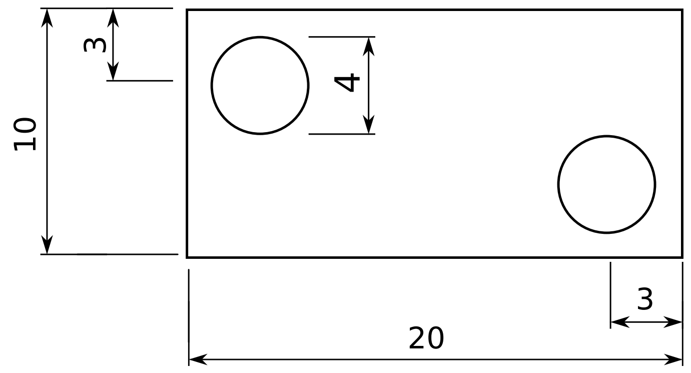


Figure 2.4: Geometrical setup of the numerical experiment.

The material is linear elastic, with parameters $E = 147.25$ and $\nu = 0.3$. The stiffness tensor $\mathbf{E}$ is:

$$\mathbf{E} = \frac{E}{1-\nu^2}\begin{pmatrix} 1 & \frac{1}{\nu} & 0 \\ \frac{1}{\nu} & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{pmatrix} \tag{2.5}$$

To demonstrate the convergence properties of the algorithm, the following modified von Mises criterion was used (σ_i is the i^{th} principal stress):

$$\mathcal{C}(\mathbf{s}) = \begin{cases} 1 - \frac{\sigma_c\sqrt{2}}{\sqrt{(\sigma_0-\sigma_1)^2+\sigma_0^2+\sigma_1^2}} & \text{if}\sqrt{\left((\sigma_0-\sigma_1)^2+\sigma_0^2+\sigma_1^2\right)/2} > \sigma_c \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

A material which can only fail along shear planes was also tested:

$$\mathcal{C}(\mathbf{s}) = \begin{cases} 1 - \frac{\sigma_c\sqrt{2}}{\sqrt{(\sigma_0-\sigma_1)^2}} & \text{if}\sqrt{\left((\sigma_0-\sigma_1)^2\right)/2} > \sigma_c \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

And finally a Mohr-Coulomb criterion:

$$\mathcal{C}(\mathbf{s}) = \begin{cases} 1 - \frac{\sigma_c}{\max|\sigma_i|} & \text{if } \max|\sigma_i|, > \sigma_c \; i = 0, 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

With the critical value $\sigma_c = 23$ in all cases.

A displacement is imposed on the upper and lower boundaries, such that complete failure of the specimen does not occur. Here the imposed displacements are 1 and -1.

2.3.2 Governing Equations

This damage model was developed for complex microstructures, with individually simple materials. Typically those are simple elastic or visco-elastic materials to which are associated a damage model and a fracture criterion. The tests were performed on a damaging linear elastic material.

The classical equation for elasticity relating stress, σ, with strain, ϵ is:

$$\sigma = \mathbf{E} : \epsilon \tag{2.9}$$

Irreversibility is introduced as a modifier of the original Cauchy-Green stress tensor $\mathbf{E}_0$

$$\mathbf{E}_{\text{eff}} = \eta(\mathbf{E}_0, \epsilon, t, \ldots) = E_0 \cdot \eta(\mathbf{s}) \tag{2.10}$$

where η is a tensor function of the current state and history of the material. All the elements are in the [0,1] range. The functions considered are of the form:

$$\mathbf{E}_{\text{eff}} = \mathbf{E}_0 \cdot \eta(\epsilon, t, \ldots) \tag{2.11}$$

where $\cdot$ denotes the Hadamard (i.e. term by term) product, $\mathbf{s}$ is the material state, current and history. This restriction has as a consequence that the following expression can be written by taking the differential of the state:

$$\mathbf{E}_{\text{eff}}(\mathbf{s} + \mathrm{d}\mathbf{s}) = \mathbf{E}_0 \left(\eta(\mathbf{s}) + \mathrm{d}\eta(\mathbf{s})\right) \tag{2.12}$$

Where d denotes the differential. Irreversibility is written as functions affecting the components of the Cauchy-Green stress tensor.

The integral of the stress over the domain are the forces $\mathbf{f}$, internal and external, and ϵ is, assuming small strains:

$$\epsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) = \nabla_s \otimes \mathbf{u} \quad \forall i, j = 1, 2, 3 \tag{2.13}$$

With u_i are the displacements, and x_j the spatial coordinates. Equilibrium is written as (in the absence of body forces):

$$\nabla \cdot \sigma = \mathbf{f} \tag{2.14}$$

With $\mathbf{f}$ the internal and external forces. Thus, inserting Eq. 2.14 in Eq. 2.9:

$$\mathbf{f} = \nabla(\mathbf{E}_{\text{eff}} \nabla_s \mathbf{u}) \tag{2.15}$$

Now, the integral form of the equation can be written over the material domain Ω, choosing $\delta\mathbf{u}$ in the subspace of kinematically admissible functions $\mathcal{V}$, u is in the function space $\mathcal{U}$:

$$\mathbf{f} = \int_{\Omega} \left(\nabla\left(\mathbf{E}_{\text{eff}} \nabla_s \mathbf{u}\right) \delta\mathbf{u}\right)\, \mathrm{d}\Omega \qquad \forall\{\mathbf{u} \in \mathcal{U},\ \delta\mathbf{u} \in \mathcal{V}\} \tag{2.16}$$

Integrating by parts, and choosing the test function in the same subspace as $\mathbf{u}$:

$$\mathbf{f} = \int\limits_{\Omega} \left(\nabla^T \mathbf{u} \left(\nabla_s \mathbf{E}_{\text{eff}}\right) \delta\mathbf{u} + \nabla_s \mathbf{u} \mathbf{E}_{\text{eff}} \nabla_s^T \delta\mathbf{u}\right) \, \mathrm{d}\Omega \qquad \forall\{\mathbf{u}, \delta\mathbf{u} \in \mathcal{H}^2\} \tag{2.17}$$

Finally, if $\nabla_s \mathbf{E}_{\text{eff}}$ is negligible over Ω, the usual weak form of linear elasticity is derived.

$$\mathbf{f} = \int\limits_{\Omega} \left(\nabla \mathbf{u} \mathbf{E}_{\text{eff}} \nabla^T \delta\mathbf{u}\right) \, \mathrm{d}\Omega \qquad \forall\{\mathbf{u}, \delta\mathbf{u} \in \mathcal{H}^2\} \tag{2.18}$$

With $\mathcal{H}^2$ a Hilbert space.

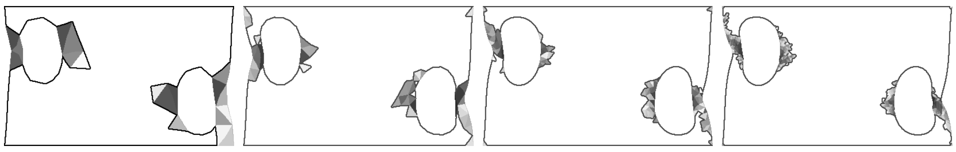
The derivation of the discrete form is done by choosing a suitable space in which to approximate the displacement. Typically the Lagrange polynomial spaces $\mathcal{L}^i$ are used in finite elements because they simplify post-processing. A compact functional basis is associated to the degrees of freedom of the elements. If h_i is the i$^{\text{th}}$ basis function, the discrete problem corresponding to Eq. 2.18 in element e is [79]:

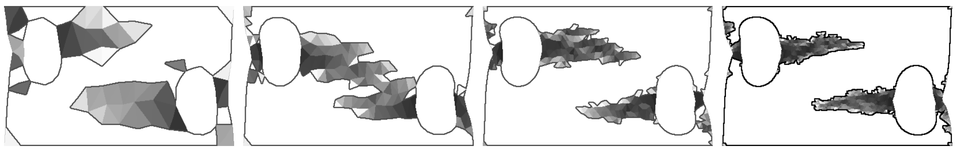
$$\underset{i,j}{\mathsf{A}} \left(\int_e (\nabla_s h_i) \mathbf{E}_{\text{eff}} (\nabla_s h_j)^T \, \mathrm{d}e \right) \mathbf{u} = \mathbf{f} \tag{2.19}$$

Where A denotes the assembly of submatrices.

2.3.3 Results

As the mesh is refined, the damaged area does not shrink, its shape becomes more precisely defined as the number of elements increases. This algorithm avoids crack coalescence on a single band of elements, and does not exhibit pathological mesh dependency. Most important, the energy dissipated during deformation converges with mesh refinement.


(a) von Mises fracture criterion


(b) Mohr-Coulomb fracture criterion

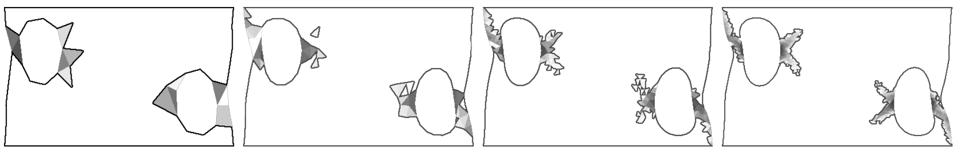

(c) Failure only along shear planes

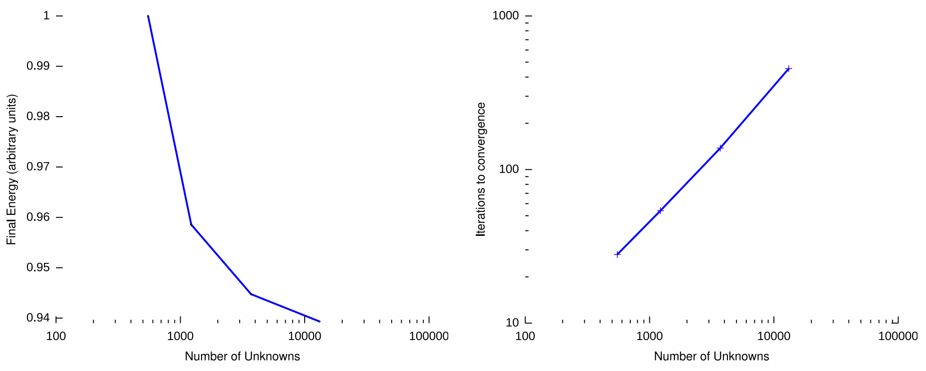
Figure 2.5: Damage localisation with successive refinements of the mesh. Damage is greyscale-coded from black, 1, to light grey, undamaged material is white. This figure shows how the damaged surface stays relatively constant as the mesh becomes finer. The effect of choosing different fracture criteria is apparent. The more ductile von Mises (a) produces more spread-out damage patterns, whereas Mohr-Coulomb (b) leads to the apparition of crack-like patterns. A material which can only fail along shear planes is also shown as an example (c).

This result highlights the ability of the algorithm to spread damage as non-local smoothed approaches without the need to specify explicitly a distribution function for the damage. The shape of the area damaged emerges from the fracture criterion, and not from an *a priori* guess.

Fig. 2.5 shows the simulation. Each row shows a different yield criteria (von Mises, Mohr-Coulomb, pure shear failure), within the rows the mesh refinement increases from left to right. The damage pattern is strongly affected by the choice of failure criterion, but not much by the successive mesh refinements. The precision of damage attribution is limited by the mesh fineness, but this example highlights the good convergence in terms of geometry of this algorithm.

The number of steps required by the algorithm to converge to a state of the material where an equilibrium is reached depends on the setup of the problem, the size of the ball considered for the comparison between elements, the number of elements and the damage increment. For the simple homogeneous case presented above, convergence was studied with respect to the number of unknowns.

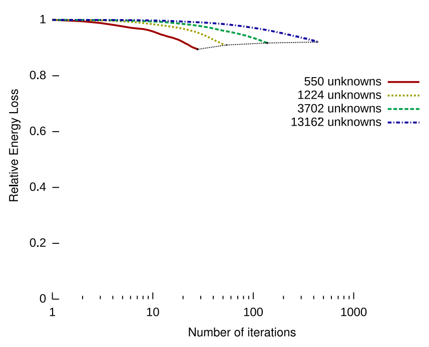
The elastic energy at convergence for the same meshes was studied. Both the relative and absolute energy losses converge as the number of elements is increased. There is a distinction between absolute and relative energy, because the absolute energy at the beginning of the iteration comes from the varying discretisations of the problem: the pores are not satisfactorily when the mesh is coarse.



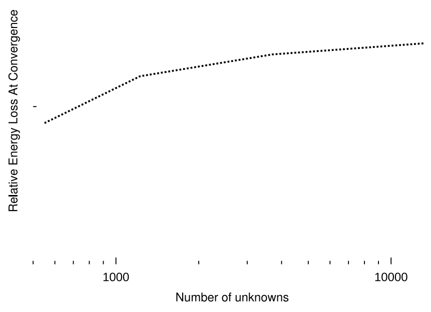
(a) Absolute Energy **(b)** Algorithm Convergence

Figure 2.6: (a) Convergence of the absolute energy stored in the system with increasing number of unknowns. The absolute energy is different also because of different discretisations of the sample geometry. (b) Log-log plot of the iterations to convergence and the number of unknowns. The order is $\mathcal{O}(n^{1.72})$.

The algorithm exhibits a characteristic convergence pattern in terms of energy release. This pattern depends on the successive relaxations and is characteristic of the geometry of the sample considered.



(a) Convergence with mesh



(b) Final energy

Figure 2.7: (a) Energy release during the iteration. The final energy release is indicated by the small dotted black curve (compare to the plot on Fig. 2.6a for absolute values). Similarly shaped curves indicate that the energy releasing process goes through the same states in different meshes. (b) Convergence of the relative energy stored in the system with increasing number of unknowns. The shape of the curve (in log-log) indicates that the energy value converges faster than the element size reduction.

2.4 Discussion

The algorithm outlined in this chapter was developed to solve composite problems with explicit geometry. It is energetically correct and captures the shape of damage domains without *a priori* knowledge other than the material law. The material law is expressed as a series of states corresponding to damage increments.

The algorithm described hinges on the description of the material behaviour as irreversibility occurs. Localisation of the damage, in the sense that micro-defects coalesce into a macro-crack, is implicitly described by the damage state update: a cohesive law can be introduced in the material behaviour there. This algorithm can thus also be used to make the transition between continuum and discontinuous techniques such as the XFEM: the introduction of a localised discontinuity in an element can be considered as a further damage state.

The damage increments can be adjusted to each problem to obtain the best performance. As a direct simulation, this algorithm cannot be guaranteed to be absolutely stable. If the increments are too large, the softening of the material at the local level will not be adequately captured. However, if the increments are too small, convergence will be very long to obtain. However we experienced no unstable behaviour in our experiments.

Crucially, this algorithm converges with mesh refinement. This is necessary when definite damage paths are sought, for example in cases where damage is both present throughout the material and highly localised. The ASR example showed how such properties are required for the validation of a microstructure-level degradation mechanism. This property is also useful when precision is only required at some places: the usual mesh-refinement heuristics used in mechanical simulations will also result in better damage computation.

The boundary conditions are assumed to be varying monotonously during a loading step. Thus, this algorithm is ill-suited to capture snap-back: at each step of the simulation, equilibrium will be reached for the set boundary conditions. An extra loop in the algorithm would be required to keep the load at equilibrium.

Although the algorithm is inefficient when damage is concentrated on a few paths, it is very efficient when widepread damage occurs in complex microstructures, making it very well suited for the simulation of degradation mechanisms at the meso-level in heterogeneous concrete-like composites.

The implementation of the algorithm is also appropriate for a space-time implementation: the damage can be expressed as a continuous function over the time slice and can be increased in a manner similar to the quasi-static case. This opens the possibility of computing damage in visco-elastic materials.

2.5 Conclusion

A direct algorithm for the computation of damage states in materials has been presented. This algorithm can be applied to cases where load depends on the varying geometry of the sample, and can be used in composite cases. The convergence of the algorithm in terms of energy and domain definition was studied. Applications to simple and complex cases were presented, highlighting the possibilities of using the algorithm to study durability issues, notably in cementitious composites.

Future work involves the extension of the implementation to XFEM crack initiation and propagation as well as the implementation of more complex damage models, using the same element-ranking approach to compute the locus of the crack initiation. This algorithm should also be implemented in space-time and tested in visco-elastic cases, using damage functions to form materials which are functionally graded in time.

Part II

The Alkali-Silica Reaction

CHAPTER 3

Experimental Programme

3.1 Introduction

The alkali-silica reaction (ASR) causes macroscopic expansion of concretes made with reactive aggregates. The curve of expansion resembles a sigmoid, three stages frequently described as initiation, reaction and exhaustion. The effect of some parameters on the kinetics of expansion is well identified while others are less well understood. For example, the reaction follows an Arrhenius law with temperature. The effect of some variables linked to the concrete formulation, such as aggregate size and applied stress are less clear.

To understand the links between the macroscopic effects – expansion, mechanical properties – and the micro-structural causes, an experimental programme was put in place. The principal objective was to understand the micro-mechanical aspects of the reaction, so a protocol had to be designed which would minimise the variations caused by alkali leaching but also paste hardening and autogenous shrinkage, while still accelerating the reaction enough for laboratory testing purposes.

The goal of understanding the microstructural consequences of the reaction is to be able to formulate a model with enough predictive power to be of use in engineering applications. This thesis follows the work of Ben Haha, who identified a key link between damage at the microstructure level and expansion at the macroscopic level. The experimental programme set up first verified the generality of the findings. Further experiments were performed

Table 3.1: Overview of the experimental programme. Any aspect of the experimental protocols exposed in this chapter which might be unclear can be obtained by asking the author or the members of the LMC.

Experiment	Samples	Comment
Effect of D_{max}. Samples cast with the SI aggregate.	11	$7\times7\times28$ cm prisms for 0-8 mm and 0-20 mm , Cylinders with embedded sensors for 0-16, $4\times4\times16$ prisms for mortars.
Fractions. Samples cast with SK and SI fractions.	16	$7\times7\times28$ cm prisms, 3 with inlets, 1 without, sliced for microscopy per condition.
Casting direction. Single 30 cm cube cast with SI and cored.	10	Cored cylinders, 6 used for expansion.
Generality. Samples cast with all the aggregates. Additional samples cast during a diploma project with the SS aggregate.	57+3	$4\times4\times16$ cm prisms per aggregates, 3 for expansion, 6 for mechanical tests/microscopy, $7\times7\times28$ prisms for additional testing of control aggregate.
Effect of load. Samples cast with the SI and SK aggregates.	12	Instrumented 20 cm diameter cylinders, 2 reactive and 1 control per load condition.
Microbars. All aggregates tested.	63	12 samples per aggregate tested.

to test the size effects of aggregates, the effect of the direction of placing and the effect of externally applied loads. The literature on these aspects is presented, followed by the materials and methods and the presentation of the experimental results.

3.1.1 Key Findings of Previous Thesis

Expansion-Reaction

In his thesis, Ben Haha showed that the expansion of mortar and concrete bars could be related to a measure of damage at the aggregate level which was interpreted as a measure of the percentage of aggregate which had reacted to form silica gel. The damage of the aggregates was measured using back-scattered electron (BSE)-image analysis (IA) on polished sections (Fig. 3.1). The damage was found to correlate to the expansion independently of the cure temperature and the amount of alkalies in the mix. Ben Haha further demonstrated that the expansions could be renormalised as a function of the aggregate content, which allowed comparison between mortars and concretes.

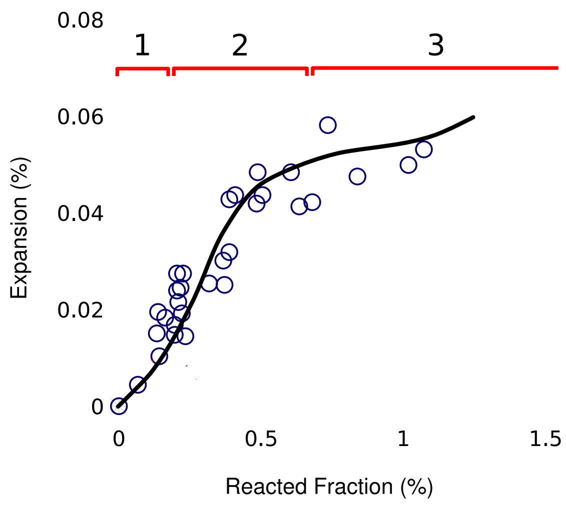


Figure 3.1: Expansion-Reaction curve as reported by Ben Haha [1, 2]. The black line is a visual guide only. The three phases which can be distinguished are the initiation phase, a propagation phase and an exhaustion or slow-down phase.

Ben Haha verified the relationship with three curing temperatures, three aggregates and three alkali content in the mixes. However, only two of the aggregates were alkali reactive and it remained necessary to verify further the results with more varied aggregates.

Damage and loss of Mechanical Properties

Ben Haha also related the loss of mechanical properties of the concrete with time and with the extent of the reaction. This provided a link between the observed degradation of the concrete and microstructural observations. This observation is crucial because it indicates that it is possible to formulate a model of the ASR based on a direct measure of the reaction and relate it to engineering properties.

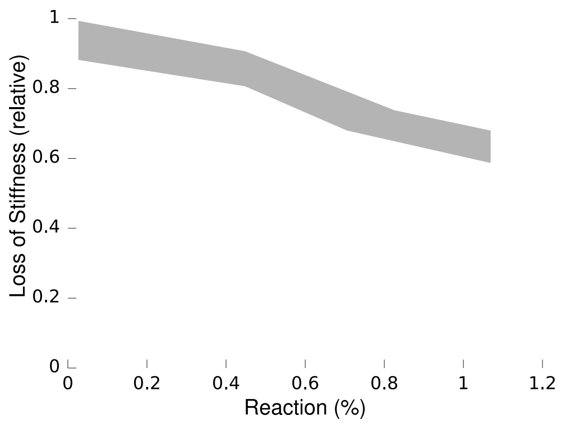


Figure 3.2: Relative loss of stiffness of a reactive sample compared to a non-reactive control. Adapted from Ben Haha [1].

This result showed that in mixes cast from slowly reacting aggregates the damage induced by ASR came from the degradation of the aggregates themselves and not by paste cracking after having been filled with gel (Fig. 3.2) as commonly assumed.

3.1.2 Programme Overview

Ben Haha concentrated on the free expansion of samples induced by ASR. The first goal of the present experimental programme was to verify the key findings with more aggregates. The damage-expansion relationship could serve as a basis for a mechanical model. However, a model applicable for engineering purposes must be able to take into account the mix design parameters, such as aggregate gradation. It must also account for boundary conditions, such as load.

The generality of the free expansion-reaction curve was checked using the same protocol as Ben Haha, but using a range of different aggregates. To perform the image analysis on the new aggregates, a custom image analysis code had to be developed.

The other parameters affecting the development of the effects of the reaction were studied using SI the most reactive aggregate Ben Haha used. Since these other parameters were studied over longer periods of time with concrete mixes, a modified protocol was designed to prevent leaching, in which samples were immersed in a solution with an alkalinity similar to the cement paste pore solution.

The parameters studied in this thesis are:

- the size effects of aggregates, both the effect of individual reactive fractions, and of the D_{max},
- the effect of the direction of placing,
- the effect of restraining stress.

Literature Background

Size Effects The effect of aggregate size and fraction has been previously studied and this has led to recommendations that specify correction factors to extrapolate from laboratory specimens to field structures. The size effect can be understood as the effect of increasing the D_{max}, or the partial contributions of the different size fractions. Both parameters have been reported to either increase or decrease the expansion, and no satisfactory mechanism has been proposed to explain the experimental observations.

Zhang and co-workers studied the influence of the large aggregate content in the concrete mix for fully reactive mixes [80]. They found that depending on the reactivity of the aggregate the overall effect could be a reduction as well as an increase of the measured macroscopic expansion. This confirmed earlier reports by French [81] who also found that the most deleterious fraction in terms of expansion was the 4-10 mm fraction. The general trend however is that the presence of larger aggregates reduces expansion at early age, and increases it later. In the Norwegian recommendations [82], the expansion from large aggregates in accelerated tests is multiplied by a factor greater than one to account for the enhanced expansion potential at later ages.

Different explanations for these observations are possible. The delayed expansion of larger reactive aggregate fractions can be explained by a diffusion process, the alkali ions taking more time to migrate to the reactive zones of coarse aggregates. However, Hobbs found that some aggregates are as permeable as a 0.71 water to cement ratio ($\frac{w}{c}$) paste [83], notably certain granites. Further, in discussing accelerated tests, Wiggum and colleagues [84] found that variations of the amount of reactive material in each class have a relatively small effect on the total expansion. This observation is incompatible with the model of fully-reactive aggregates which react on the surface. But the explanation can also be found in mechanical interactions. For example, Wiggum first considered the role of grading by comparing the relative surface of such model aggregates [85].

Modelling the coupling between apparent expansion and reaction has been attempted using explicitly represented microstructures. In her PhD Thesis, Comby studied the development of a numerical tool to model the damage in ASR-affected concrete at the meso-level [86]. She studied the feasibility of 3D microstructure generation, and noted the importance of the aggregate grading for the damage evolution, and the necessity of a homogenised model for the mortar as she did not explicitly model the smallest aggregates. This study highlighted the importance of cross-class interactions. An analytical model, based on the development of reaction rims was published by Bažant [87], which takes into account the interactions between aggregates to explain the shape of the expansion curve. Attempts to integrate the effect of individual fractions in ASR models have been made. Notably, Multon has proposed an ASR model which considers the expansion of the aggregate classes separately. He also studied the coupled effect between alkali content and aggregate content [6]. He stated that the alkali content should be normalised by the aggregate content, which we believe is in contradiction with the fundamental assumption of his model, which is that aggregates react on the surface: the aggregate content to aggregate surface ratio is highly dependent on the grading, and he therefore assumes different reactivities for each class.

Experimental work on different size fractions is mostly concerned with mortars. Poyet worked on the effect of individual fractions in his thesis [15]. He separated aggregate size fraction and performed accelerated tests on mortars where only some of the fractions were reactive. He found that very fine fractions do not cause expansion. Cyr and colleagues performed a similar study with finely ground reactive aggregates of various types, and found they all reduced the expansion [88]. The aggregate used by Poyet was a very reactive flint, and this result could be explained by a pozzolanic reaction of the fine reactive particles, as also put forward by Cyr. Ramyar and colleagues performed a similar study on mortar. They tried to determine the effect of angularity as well as size, but their “natural” and crushed aggregates are of a different mineralogical nature. The fractions with the least expansive effect were the smallest and largest, namely 0.125-0.25 and 2-4 mm [89].

In summary, the observed effected of aggregate size could be due to various causes of either chemical or mechanical origin:

- Different mineralogy of aggregates at different sizes
- Diffusion process of the alkalies to the core of larger aggregates
- The forces exerted by aggregates on their neighbours and the different crack propagation lengths in aggregates as a function of their size

All these effects are not necessarily mutually exclusive and could also happen simultaneously.

Placing Direction Anisotropy of swelling can be observed in unconstrained conditions. The origin of the anisotropy under these conditions is not well understood, but it has been reported in the literature and is consistent with the known anisotropy of concrete in terms of fracture properties [90]. Smaoui and colleagues conducted a phenomenological study of the anisotropy of ASR and found that it was related to the relative directions of casting and measurement [13], confirming earlier findings by Larive in her PhD thesis [12]. All these studies show that concrete expands more in the direction parallel to the casting than in the orthogonal directions. The typical ratio reported between the expansions is 0.6 to 0.7.

Anisotropy has also been studied in models. Some recent models used for structural analysis however do not consider any anisotropy in the reaction, but will produce anisotropic results at the sample level because they consider a diffusion process for water to initiate the swelling [91, 92]. The diffusion properties of concrete are assumed to be anisotropic in the model, and as the expansion is linked to the moisture content, anisotropic expansion results. Anisotropy in models can also be produced by oriented macroscopic cracking, such as in the work of Grimal and colleagues [93]. This is again a different kind of anisotropy, which is due to the member geometry. Multon introduces anisotropy as an external correction factor [8]. In general, anisotropy in models is of a phenomenological nature, because its physical cause is not understood. More frequently however, anisotropy is discussed in relation with confining stress.

Restraining Stress Mohamed and co-workers studied the effect of confinement by wrapping ASR-reactive samples in carbon-fibre coatings. They found that restraint reduced the measured expansion, but did not assess the microstructural consequences. They also found that the stress required to reduce expansion was lower than that predicted by earlier probabilistic modelling [14]. A study of Multon and Toutlemonde shows the restraint has a directional effect: it affects the concrete differently depending on the casting direction [8]. In a work relating theoretical models to ASR-affected reinforced concrete, Winnicki shows that the coupled effect of mechanical degradation and restraint is difficult to model using continuum approaches, and fails for high levels of reinforcements [94]. This study highlights the importance of understanding the failure modes of the material for the construction of ASR models.

The effect of stress itself is also discussed in terms of its apparent moderating effect on the reaction. Léger and co-workers proposed a methodology to model the effect of ASR on a dam, and amongst other factors, they introduced a correction function for stress, where the expansions considered varied from free expansion (stresses below 0.3 MPa) to fully contained expansion (stresses above some value lying between 5 and 10 MPa) [4]. This approach was still used by Capra *et al.* [95] ten years later. Herrador and colleagues, in a study on dam concrete reviewed various previous experimental works describing the existence of an “inhibition pressure” which would prevent expansion [96]. This pressure is found to lie, with considerable variation, between 3 and 10 MPa. Binal measured the free pressure exerted by artificial gels and found it to lie in the 0.1 to 2.7 MPa range [9]. This measure is consistent with the stress reported by Ferraris and colleagues which compared reactive and non-reactive samples in strain-restraining frames [97]. Using scanning electron microscopy (SEM) to study specimens, Larive concluded that the pressure does not inhibit the reaction but only the apparent expansion [12]. However, she identified the reaction by looking for “reaction rims”[1].

When stress is applied to a sample, the latter’s volume is proportionately reduced due to Poisson’s ratio. Further, in long-term experiments the paste will creep, further reducing the volume. ASR tends to increase the volume of a sample, by replacing aggregate mass by a water-filled silica gel of higher volume. It increases the apparent volume further by inducing cracks in the aggregates and paste. The range of stresses known to apparently inhibit the reaction is significantly below the typical paste and aggregate compressive strengths, and itself induces no irreversible damage. However, crack propagation direction is in general strongly affected by the direction of the principal stresses. We can therefore formulate the following hypothesis: the imposed stress does not affect the volume increase directly linked to the reaction, but it affects the damage induced by forcing a preferential propagation direction for cracks. As a consequence, once the effects of creep and elastic deformation are removed, there should be no significant difference in the volumetric evolution of samples cured the same way.

Following a similar argument, any preferential orientation due to microstructural differences caused by the placing should be constant with time. If the crack propagation from the aggregates is stable, the speed of propagation should be finite and only due to the current state of the reaction. The speed of propagation of cracks in directions perpendicular and parallel to the casting direction should be proportional to each other, with a constant ratio over time. Therefore, the absolute difference in expansion in directions normal and parallel should grow linearly with time. The hypothesis that the apparent expansion inhibition can be understood as a bias in crack propagation direction was tested in this study.

[1]Reaction rims are regions of increased porosity around reacted aggregates. They are frequently thought to result from the cracking of the paste around the aggregates.

3.2 Materials and Methods

3.2.1 Materials

Two groups of aggregates were tested. The first is a set coming from North America where they have been extensively tested in large-scale studies (Tab. 3.2). This group is composed only of aggregates which are known to be reactive. They serve two purposes: first, the link between their expansion and the measure of their reaction is compared against the original results of Ben Haha, and second, they serve as a control for the cure protocol used. The second set of aggregates comes from quarries in Switzerland and was tested for reactivity (Tab. 3.2). Of these, the SS was studied in a separate diploma thesis [98].

Table 3.2: Aggregates from North America and Switzerland.

Aggregate	Mineralogy type
AS	Mixed gravel – much used for the study of AAR.
AI	Granitic gneiss, metarhyolite – Thought to be similar to the aggregates studied by Ben Haha.
AL	Homogeneous quartzite
AJ	Reactive river aggregate
AW	Very reactive
SA	Mixed river aggregate
SF	Mixed river aggregate
SS	Mixed mineralogy reactive Alpine aggregate
SI	Mixed mineralogy reactive Alpine aggregate used by Ben Haha
SK	Mixed mineralogy Alpine aggregate

All aggregates were first tested following the MICROBAR test which is used to classify the aggregates as potentially reactive or not [99]. This test is part of the NF XP P 18-594 norm from the Agence Française de Normalisation (AFNOR). The results are reported on Fig. 3.3. All aggregates were found to be potentially reactive (including the non-reactive one) and this test was found to be undiscriminating. Notably, no link between expansions in the MICROBAR test and the expansions obtained in the accelerated mortar bar test was observed.

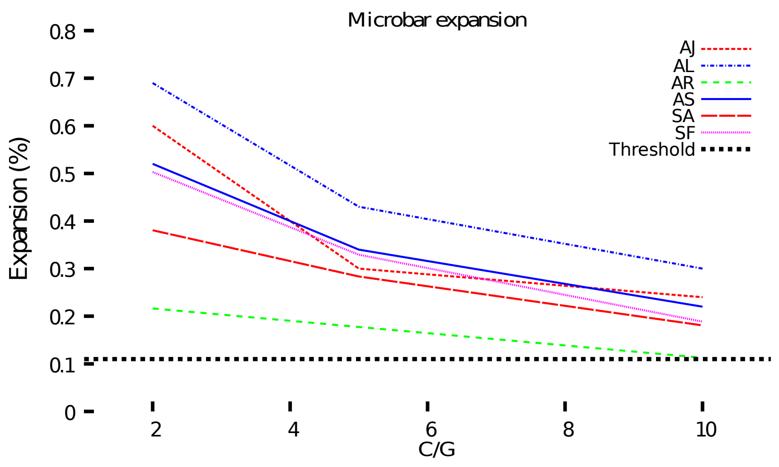


Figure 3.3: MICROBAR results for most aggregates used in the study. All aggregates are above the reactivity limit.

The experiments designed to study the effect of the individual fractions required additional preparation of the aggregates. To ensure that there would be no effect from different particle size distribution (PSD), the fractions of the reactive aggregate were recomposed to match those of the non-reactive control. The reactive aggregates were prepared by crushing and sieving. All sub-fractions were kept separated and individually weighted, a list of which is provided in Tab. 3.3. As all reactive fraction were prepared from aggregates originally from the same class, the fraction of reactive material in each class is constant.

Table 3.3: List of sieves used for the separation of the reactive aggregates. The separations indicate the cut-off points for the general classes used. The percentages reported are for the SK classes.

Opening in mm	0-4	4-8	8-16	16-24
0.053	3.5	1.5	1.2	0.8
0.063	4.7	1.5	1.2	0.8
0.10	7.1	1.5	1.2	0.8
0.16	10.0	1.5	1.2	0.8
0.25	14.1	1.5	1.2	0.8
0.40	22.1	1.5	1.2	0.8
0.63	34.1	1.5	1.2	0.8
1.0	49.0	1.5	1.2	0.8
1.6	64.2	2.8	1.2	0.8
2.5	80.5	4.4	1.2	0.8
4.0	97.8	14.3	1.2	0.8
5.0	100.0	40.6	1.2	0.8
6.3		66.7	1.5	0.9
8.0		91.3	3.6	1.1
10.0		99.7	20.9	1.7
12.5		100.0	66.3	3.9
16.0			97.2	26.7
20.0			100.0	62.9
25.0				100.0

The non-reactive aggregates were available already in 0-4, 4-8, 8-16 and 16-32 mm classes. We produced a 16-24 mm class by sieving the 16-32 mm class using a 24 mm sieve.

The PSD of all non-reactive fractions was recorded, and identical fractions were recomposed from the reactive sub-fractions. The initial PSD of all fractions is reported on Fig. 3.4. Once the aggregates had been prepared for a batch, they were shuffled 8 times using a separator to ensure a homogeneous blend.

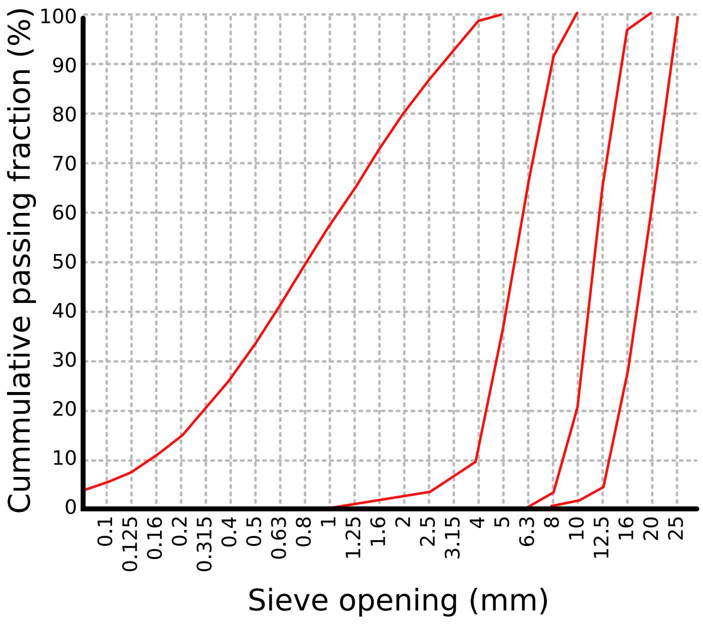


Figure 3.4: PSD of the aggregate fractions used.

The PSD of the mix design was then composed to best approach the Bolomey reference curves [100], using either all reactive or partly reactive fractions (Fig. 3.4).

The $\frac{w}{c}$ was set at 0.43 and, to improve fluidity, 1% in mass of Rheobuild 5500 was added. Further, alkalies where added to the mix water after it had been weighted so as to bring the initial alkali content of the mix to 0.8% sodium oxide equivalent (Na_2O_{eq}) of the mass of cement. A table of all samples cast is provided below (Tab. 3.4).

Table 3.4: Mixes from partly reactive aggregates.

Fraction	0-2	2-4	4-8	8-16
Reactive	**yes**	no	no	no
Reactive	no	**yes**	no	no
Reactive	no	no	**yes**	no
Reactive	no	no	no	**yes**

The cement used was a Pur 4 low alkaline cement from Holcim with 0.5% Na_2O_{eq}. All experiments were run with the cement coming from the same

Table 3.5: XRD analysis of the cement used in the expansion experiments.

Name	% Weight
Alite	55.3
Belite	16.9
Ferrite	5.7
Aluminate	9.3
Lime	0
Periclase	0
Gypsum	0
Hemihydrate	1.6
Anhydrite	4.2
Quartz	0
Portlandite	0.5
Calcite	5.7
Arcanite	0.7

batch. The result of an X-Ray diffraction (XRD) analysis of the cement are reported in Tab. 3.5. This cement was chosen so the alkali content of the initial mix could be controlled with an external addition of sodium hydroxide. The additional sodium hydroxide was added in the mix water to reach 0.9% of the cement mass in alkalies.

3.2.2 Methods

The same mix design was used for all the mortar bar tests. The superplasticiser would not have been necessary in all cases but was always added for consistency.

The North American aggregates were prepared following the same accelerated mortar bar test modified from the American Society of Testing and Materials (ASTM) C 1260 [101] as Ben Haha so the results would be comparable. The protocol is reported on Tab. 3.7 and the mix design on Tab. 3.6. However, rather than storage in 100 % relative humidity conditions, the samples were stored in simulated pore solutions.

In 100 % relative humidity conditions it is difficult to avoid leaching due to condensation and the results show a high variability due to difficulties in controlling the saturation state.

In the first tests, the samples were kept in small quantities of water, however it was not possible to prevent leaching from occurring. Therefore

Table 3.6: Mix design for the mortar bars. The aggregates were all prepared so the 0-0.0125 mm class had been removed.

Material	Note	Quantity
Cement	Holcim Pur 4	1263 g
Water	Tap water	581 g
Aggregate		3789 g
Sodium Hydroxide	mixed with water to reach $Na_2O_{eq} = 0.9\%$ cement mass	5 g
Superplasticizer	Rheobuild 5550	10 g

Table 3.7: Curing and measuring protocol used for the mortar bars

Condition	Protocol
Mixing	• Mix aggregates and cement for 1 min at slow speed • Add water + NaOH for 1 min at slow speed • 2 min at high speed, hand mix with spatula • 2 min at high speed • Half-fill moulds, 60 hits • Fill moulds, 60 hits • Smooth surface, cover, place in humid conditions
Early cure	De-mould at 24 hour
Cure	Place in tap water at 38 °C
Measure	Wipe and measure while hot

for longer tests with concretes, the curing protocol was modified by adding NaOH to the cure water to reflect the concentration in the cement pore solution ($0.150\,\mathrm{mol\,l^{-1}}$). The modified protocol is described in the following table (Tab. 3.8). New experiments with the same aggregate as Ben Haha showed that this indeed limited the leaching (Fig. 3.5).

Table 3.8: Comparison between the first and the final protocol. The longer initial cure in the final protocol is intended to minimise paste hardening during the development of ASR.

Condition	First protocol	Final protocol
Mixing	Mortars: • Aggregates and cement for 1 min at slow speed • Add water + NaOH for 1 min at slow speed • 2 min at high speed, hand mix with spatula • 2 min at high speed • Half-fill moulds, 60 hits • Fill moulds, 60 hits • Smooth surface, cover, place in humid conditions	Concretes: • Aggregates and cement for 1 min at slow speed • Add water + NaOH for 1 min at slow speed • 2 min at high speed • Half-fill moulds, vibrate • Fill moulds, vibrate • Smooth surface, cover, place in humid conditions
Early cure	De-mould at 24 hour	De-mould at 24 hour, place in fog room until 28 days
Cure	Place in tap water at 38 °C	Place in 0.150 $\mathrm{mol\,l^{-1}}$ alkaline solutions at 38 °C
Measure	Wipe and measure while hot	Cool-down 24 hours in fog room, wipe and measure

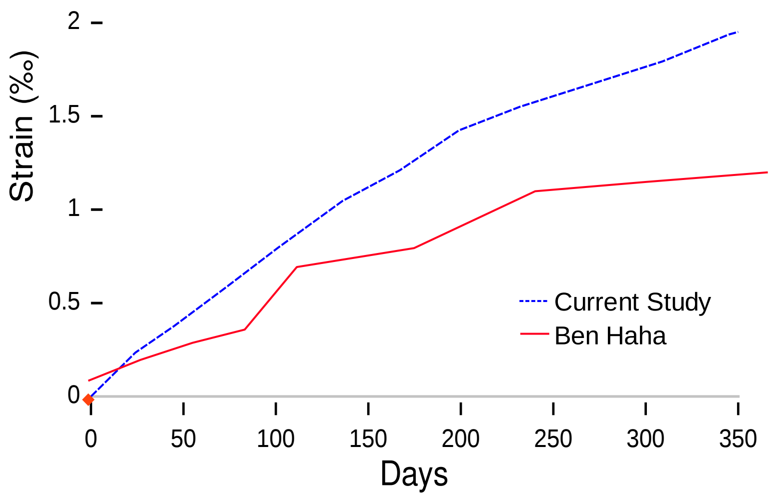


Figure 3.5: Compared expansion of concretes using the new protocol and the one from Ben Haha. The aggregate (SI) and mix designs are the same.

To better understand the coupled effects of casting direction and confinement, two complementary sets of experiments were carried out. In a first set, a cube was cast, and cores were extracted in directions normal and perpendicular to the placing direction. The cores extracted from the cube were prepared with steel inserts. In a second set of experiments focused on confinement, samples instrumented with normal and parallel embedded sensors were cast and placed in creep frames and subjected to 0, 5, 10 and 15 MPa loads. Using these setups, we could study the coupled effects of the intrinsic anisotropy of the reaction, and the redistributive effect of confining stress.

The concretes were prepared according to the protocol outlined above. However, the concrete was placed using a vibrating table for the cylinders destined for the creep frames instead of a needle. The moulds from the latter were pre-instrumented, with the sensors held by steel wires. Once filled, the moulds were placed on a vibrating table and the wires removed (Fig. 3.6). A preliminary set of samples had been prepared before using the same protocol. The sensors remained into place when this procedure was used, and no defects of the concrete were observed near the sensors in the preliminary samples.

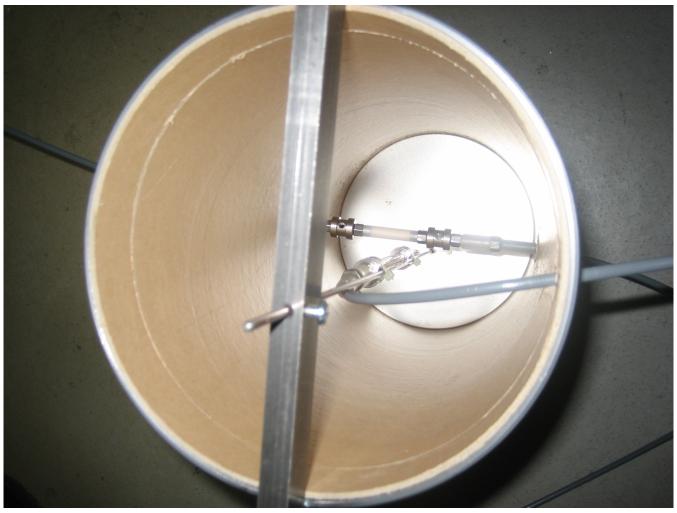


Figure 3.6: Instrumented mould prepared for the casting.

All samples, cube and cylinders, were then kept for 28 days in a fog room at 20 °C after being placed. The cube was then cored in directions normal and parallel to the direction of placing, to produce cores of 78 mm diameter, which were rectified to 250 mm in length. Afterwards all samples were stored in a temperature-controlled room at 38 °C. The creep frame was modified by adding stainless steel supports and Plexiglas vessels so the cylinders would be cured in immersed conditions (Fig. 3.7).

Four columns were prepared in creep frames. In each column, were placed a non reactive sample (on top) and two reactive samples (on the bottom). The load was kept constant in each column by hydraulic pressure. The expansion was monitored in real time using a computer which read the sensor values every 20 minutes. The non-reactive samples allowed us to extract strains due to creep from the expansion induced by ASR.

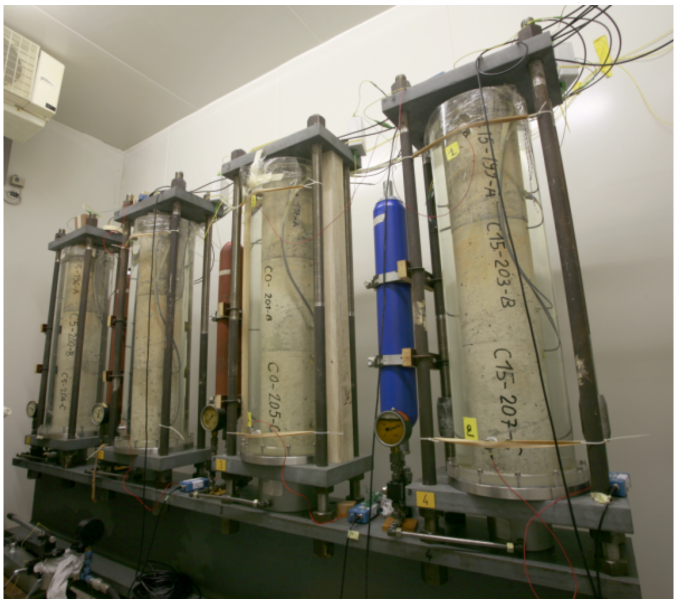


Figure 3.7: Creep frame modified so the samples are immersed. The Plexiglas vessels are visible.

The sensors embedded in each sample placed in the loading frame were:

- A longitudinal strain gauge also containing a temperature sensor. The elongation is measured using a Bragg grated optical fibre. Part of the optical fibre containing the Bragg grating is arranged in a tube. The grated part is in tension between the two ends of the tube. The ends of this part are fixed to the ends of the tube, and the tube is rigidly fixed to a host material.
- A radial strain gauge which functions on the same principle.

A schematic representation of the setup is reported in Fig. 3.8. The temperature sensor allows the correction of small strain fluctuations due to variations of the ambient temperature. The coefficient of thermal expansion was calibrated from the strains reported at 20 °C and 38 °C before the stresses were applied. The coefficients of thermal expansion found were different whether the samples were prepared with the non-reactive aggregates ($\alpha = 0.0011\,\%\,\mathrm{K}^{-1}$) or with the reactive ones ($\alpha = 0.008\,\%\,\mathrm{K}^{-1}$).

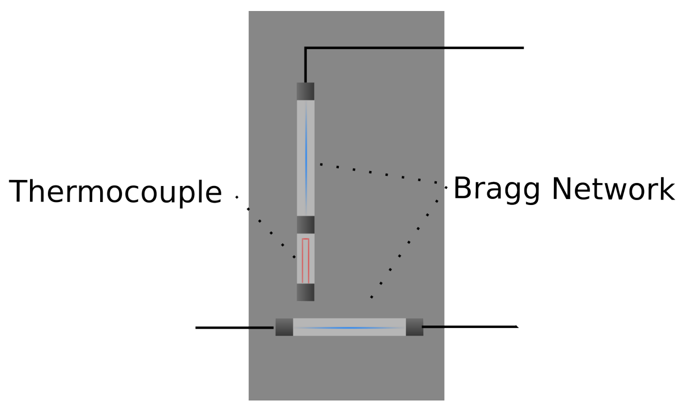


Figure 3.8: Schematic representation of the sensor placed in the loaded samples figured by the grey box.

3.2.3 Image Analysis

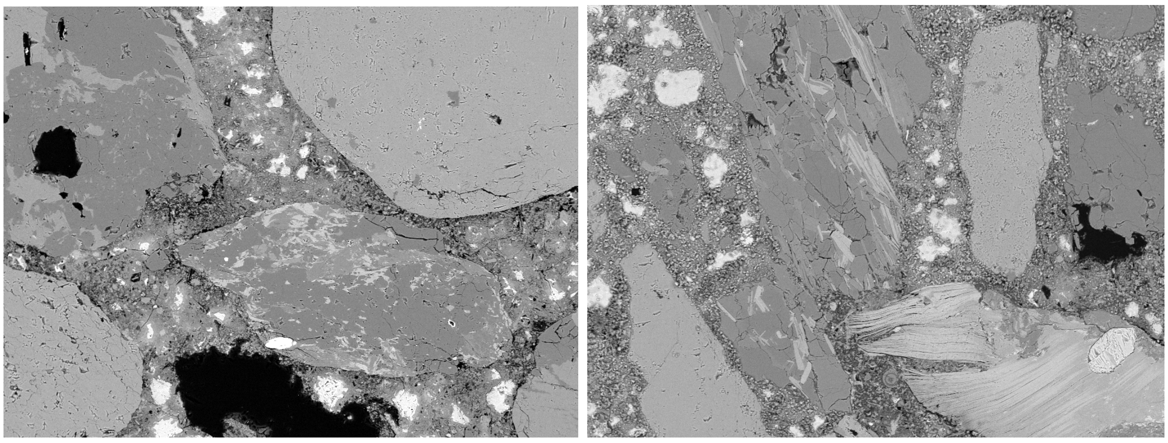
The aggregates tested are not simple quartz aggregates: visually, they are composed of different phases with grey levels overlapping with those of cement (Fig. 3.9). Notably, some aggregates appear as textured zones with some phases indistinguishable from inner calcium silicate hydrate (C-S-H). Because the aggregate phases are not easily identified with simple segmentation, an algorithm coupling segmentation with morphological selection was developed to automatically extract the aggregates.

An algorithm had been developed for the aggregates studied by Ben Haha. However, it was not general enough to be usable for the wider variety of morphologies observed in this study. Therefore the development of a new algorithm was required to be able to perform an identical analysis on all the samples we prepared.

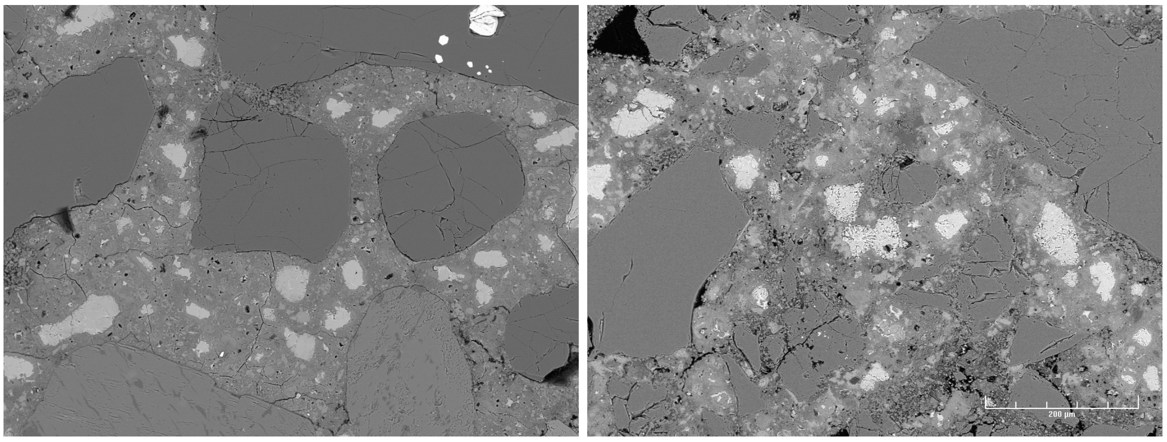
The principle of the algorithm is to perform rough segmentations based on grey levels, and then use the fast blob[2] detection [102] algorithms available in the Intel computer vision library OpenCV to classify all the features which might be aggregates[3]. Because some aggregates are composites and some have grey levels very near to the greys of paste, a two-stage classification followed by a post-treatment was implemented.

[2]A *blob* is a set of connected pixels, given a rule for deciding two pixels are connected.

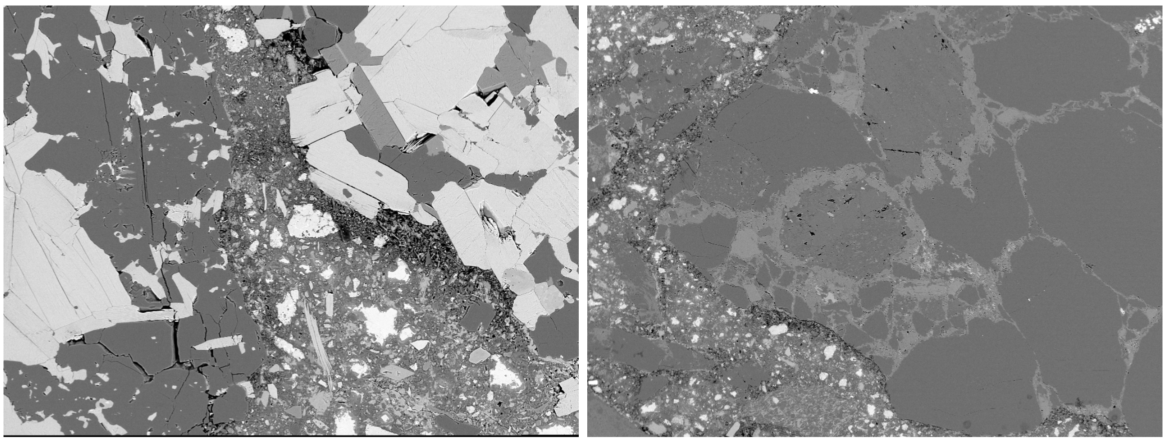
[3]The diverse nature of the aggregates and the large amounts of images prompted the development of a custom c++ code for image analysis. The code is reported in Appendix A.



(a) SA and SF



(b) AJ and AL



(c) AI and AS

Figure 3.9: Micrographs of the SA, SF, AJ, AL, AI and AS aggregates. These micrographs illustrate the wide variety in terms of shapes and textures in the selection of aggregates tested.

1. The potential aggregate parts are selected as blobs above a certain size and with a shape factor[4] below a certain threshold for greys typical of a certain mineralogical phase.

2. The potential aggregate parts are merged into a single binary map, and after a closing operation, the blob detection algorithm is applied a second time, with a larger size threshold, and a lower shape factor threshold for acceptance.

3. Finally, once potential aggregates have been identified, the largest single blob which is not an aggregate is assumed to be paste, and the identification is complete.

The first stage of classification has a double purpose. First, it consolidates the thresholded image into potential aggregate parts, thereby removing a lot of the noise from the segmentation which appears as single pixels or small clusters. Further, the blob detection algorithms selects only the outer boundary of potential blobs, which extends the potential aggregates to include their internal porosity. The second stage eliminates Portlandite clusters and inner C-S-H[5] or anhydrous grains which might have been kept after the first classification. The complete algorithm is outlined on Fig. 3.10.

The segmentation of aggregates is very efficient. A comparison was made between an automatically segmented acquisition and a manually segmented one. The relative error between the two porosity measures was 0.01%. However, to compare damage measures obtained from two acquisitions, a sensitivity analysis is required.

Images acquired using BSE microscopy can have different levels of contrast and brightness, notably between acquisitions. This means that the threshold for porosity must be set each time. Comparing different image sets becomes then subjective as this parameter is crucial for the computation of damage. To overcome this limitation, a criterion must be chosen to be able to compare the different acquisitions. This criterion must qualify the threshold grey below which an area is considered to be damaged. If the threshold is set too high, all damage will be accounted for, but there will be a high proportion of false positives. Conversely, if the threshold is set too low, only damage will be counted as damage but there will be a high proportion of false negatives.

A measure of this trade-off is the relative increase in measured damages surface per small change of the threshold. This is the derivative of sensitivity (the ratio of false positives over total negatives) with respect to retrieval

[4]The shape factor is calculated as the ratio of the area of the blob over the area of the best-fit rectangle.

[5]Increasing the curing temperature leads to the formation of a denser (and thus brighter) inner C-S-H [103]

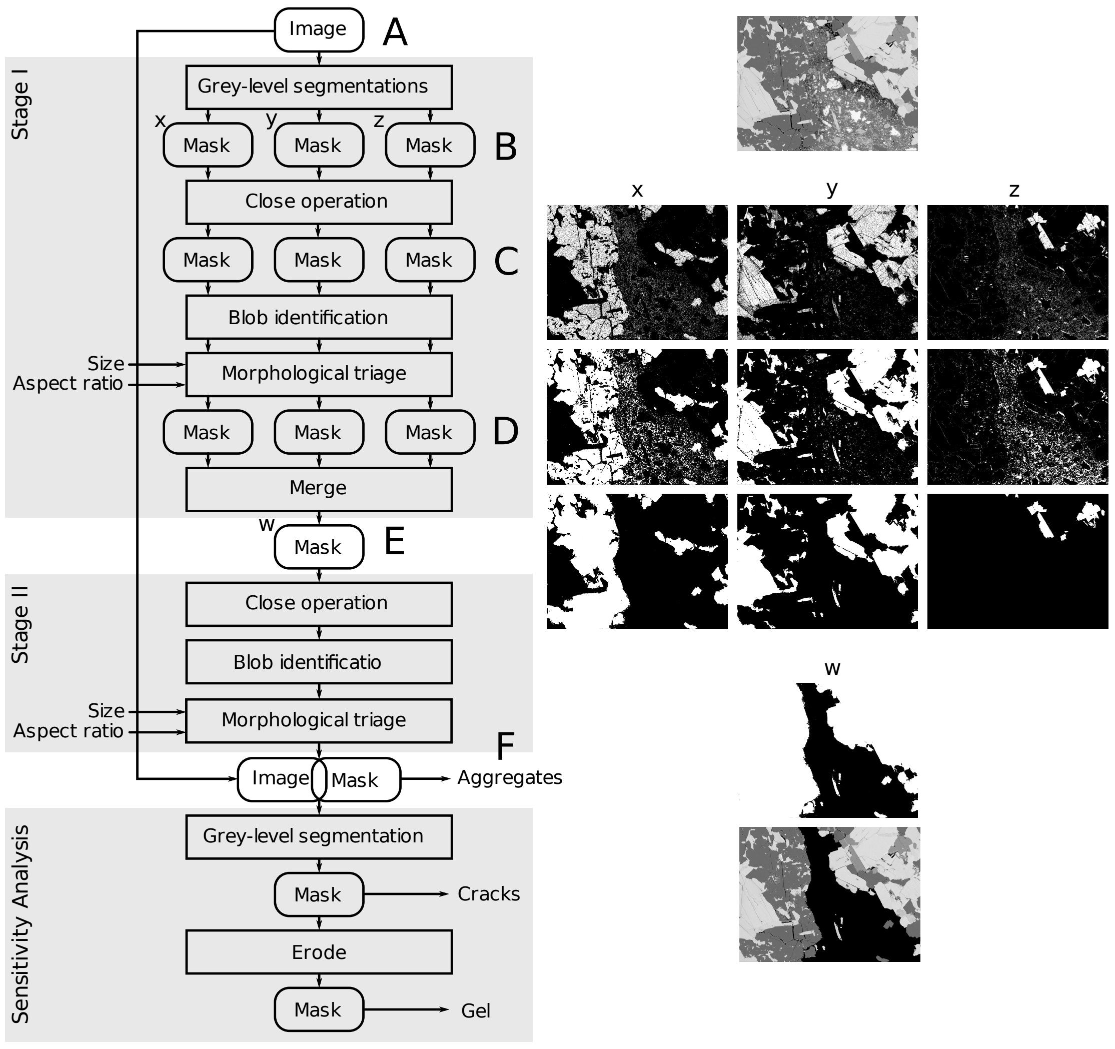


Figure 3.10: Outline of the image segmentation and analysis algorithm.

(the ratio of true positives over the total positives). An optimum where the errors are nil does not exist in general. Further, it is impossible to know the real error as a function of the parameters used for the segmentation. Although the optimum is unknown, different acquisitions can be compared if they have similar sensitivity values. Indeed, the same value means that the same trade-off was made, and therefore that the analyses are comparable. The choice of the sensitivity value is done on a single acquisition, from visual inspection.

The formula for the sensitivity calculation is:

$$\text{sensitivity} = \frac{1}{\max_{\text{damage}}} \frac{\Delta\text{damage}}{\Delta\text{threshold}} \tag{3.1}$$

A typical sensitivity curve is presented on Fig. 3.11.

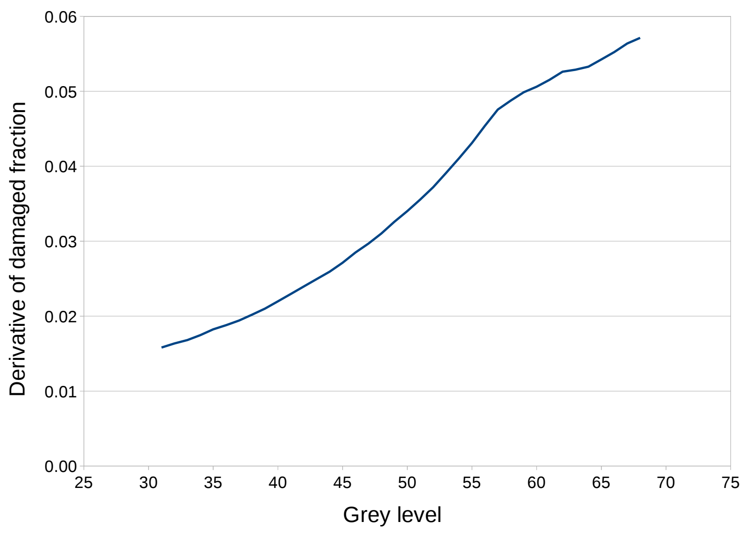


Figure 3.11: Typical sensitivity curve computed from an acquisition. The sensitivity is *in*-sensitive to the brightness and contrast of a particular acquisition and thus allows comparison between different sets of images because it measures the trade-off in terms of precision and recall for each possible threshold.

The "real" amount of cracks cannot in general be computed. It is impossible to distinguish a crack from a shadow, and the boundary between crack and aggregate is always fuzzy. Therefore, this analysis which uses the sensitivity value as a means to find comparable segmentations is the best we could find.

3.3 Results

3.3.1 Generality of the Expansion-Reaction Curve

All North American aggregates were confirmed to be expansive in the mortar bar tests. The bars were immersed in small quantities of tap water and some degree of reaction was observed[6]. However, it appears that that leaching occurred with this protocol, probably because the microstructure is still highly porous at the time of immersion. Indeed, after a fast initial expansion phase, the swelling of the samples stopped. The final expansions were lower with these aggregates, which are known to be highly reactive than with the slowly reactive aggregates used in Ben Haha's thesis. The expansion results are reported on Fig. 3.12.

Under these accelerated conditions, the expansion occurs early, while the paste is still gaining significant strength. Indeed, all expansions have levelled off before 28 days. Fig. 3.13 shows the evolution of the flexural and compressive strength for the North American aggregates. Instead of normal strength development, the mortars show effectively constant properties after only one week. Finally the properties start increasing again. No macroscopic damage was visible on the sample surfaces which indicates that the expansion is coupled with damage occurring at the microstructure level, as evidenced by the decrease in properties compared to those expected. Increased porosity in the aggregates was measured (Fig. 3.14) and could explain the observation[104].

[6]The expansions are measured from the recorded size one day after immersion.

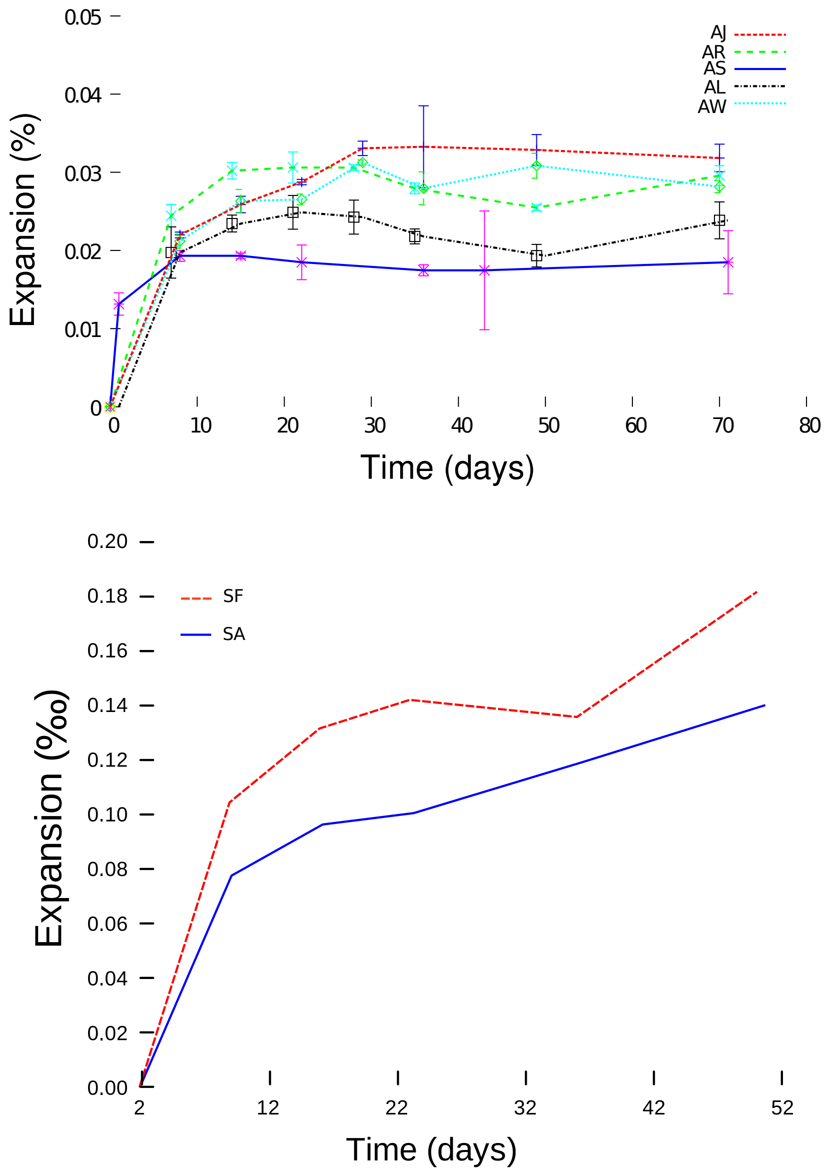
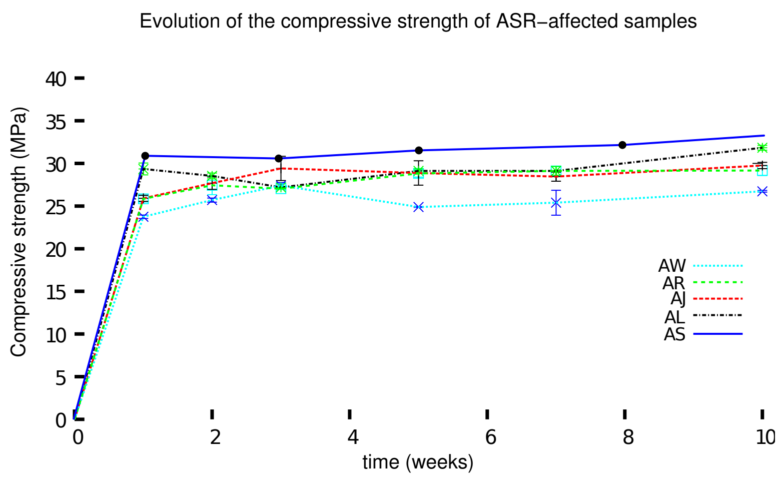


Figure 3.12: Accelerated expansion tests for the North American (top) and Swiss aggregates (bottom). All have expanded beyond the expansion threshold defined by the ASTM norm C 1260. The exhaustion of expansion can be attributed to leaching of the alkalies.



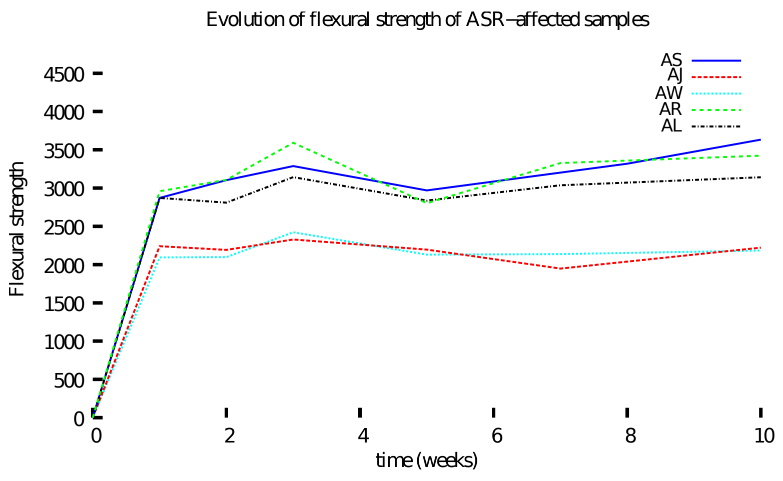


Figure 3.13: Evolution of mechanical compressive and flexural strength for North American aggregates cured at 38 °C immersed in tap water.

The data points marked on Fig. 3.14 not coming from Ben Haha were obtained from a large variety of aggregates (Tab. 3.2 and Tab. 3.2), and therefore exhibit larger dispersion. This is expected as the aggregates all have slightly different mechanical properties, as apparent from Fig. 3.13. However they still follow the same trend as that reported by Ben Haha (Fig. 3.14). This result shows that the expansion-reaction curve is not specific to the aggregates used in his study, but seems to be general for the curing conditions we used.

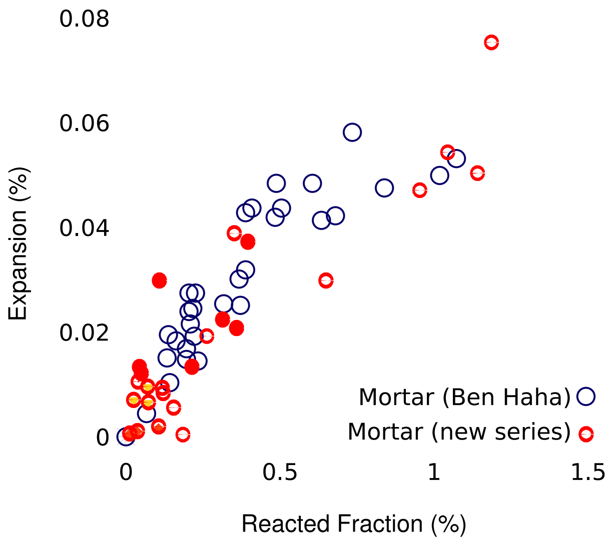


Figure 3.14: Experimental relationship between measured free expansion and measured reacted fraction. Results from Ben Haha [1], as well as from the current study are reported. The results are from acquisitions on samples mixed from North American aggregates.

3.3.2 Size Effects

The two aspects of the size effect are the D_{max} and the influence of individual aggregate size ranges. In all the experiments, the mineralogy of the aggregates is the same for all reactive fractions, and therefore does not account for observed differences in terms of kinetics or absolute differences in terms of expansions. Notably, the fraction of reactive material per unit reactive aggregate mass can be assumed to be constant in all experiments in each class.

The expansion of samples cast only with reactive aggregates are reported in Fig. 3.15. The 0-16 mm curve was measured from the unloaded cylinders in the creep frame, and therefore the sample shape is different. However, the mix and the cure were the same. The expansions reported for this sample have been corrected for the thermal expansion.

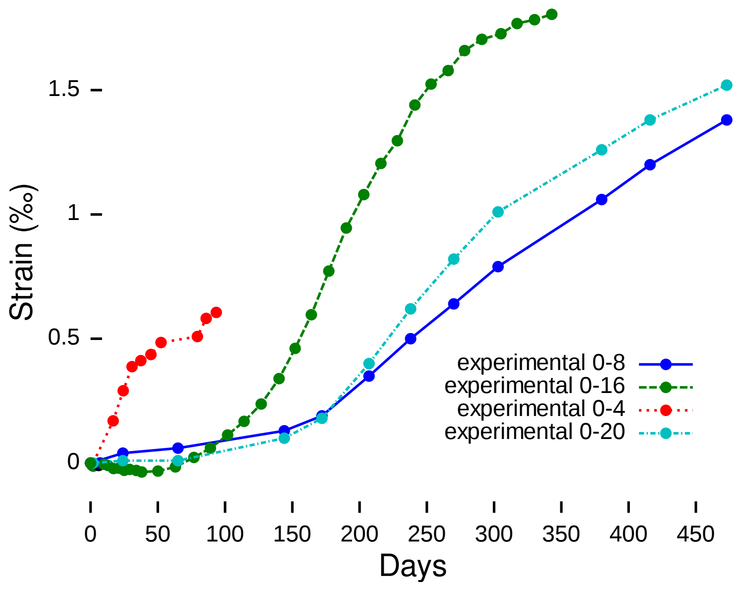


Figure 3.15: Free Expansion of specimens with fully reactive aggregates. Both the kinetics and absolute value of expansions are affected by varying aggregate size. The error of the 0-8 and 0-20 series lies within the thickness of the line.

There seems to be no monotonous trend explaining the kinetic effect of the PSD. Indeed, although the the 0-16 mm is the quickest to react, it is the followed by, in order, the 0-8 and the 0-20 mm.

To directly study the impact of individual aggregate fraction, we followed the expansion of specimen made with partly reactive aggregates. It was not possible to have a non-reactive aggregate with exactly the same mechanical properties as the reactive one. The difference in elastic properties of the two mixes was measured when samples were loaded in creep frames by comparing the relative compressions. This could induce some differences, as the homogenised stiffness of the non-reactive part of the sample varies between the experiments.

Three phases can be observed in the expansion of the partly reactive concrete (Fig. 3.16): a first phase, lasting approximately 110 days, where no significant difference is observed between the samples. Then a second phase, up to 150 days, where the larger fractions and the smaller separate. Finally, a third phase, after that where there is separation between the largest 8-16 mm, and the intermediate 4-8 mm.

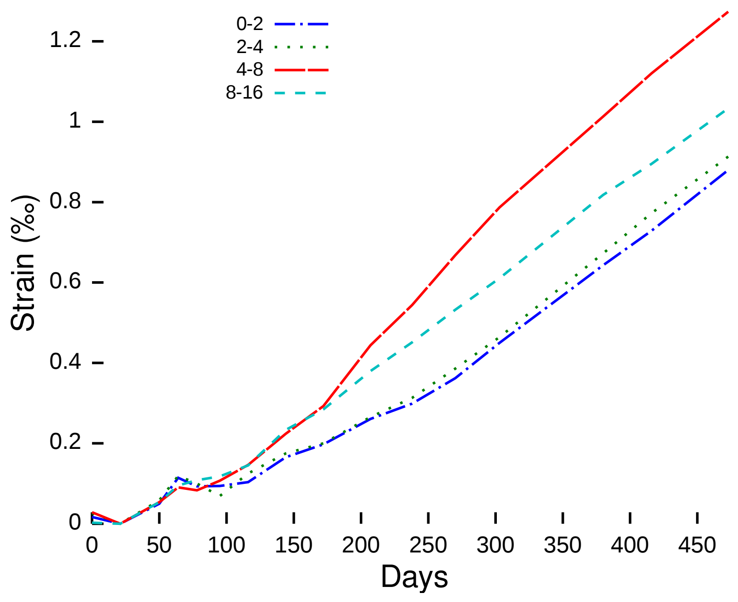


Figure 3.16: Free expansion of specimens with partly reactive aggregates.

There seems to be no significant difference in terms of expansion produced between the 0-2 and the 2-4 mm fractions. This is not consistent with previous literature reports: the smallest fraction was expected to yield less expansion. However, as reported by Dahl [82], there seems to be a threshold at 1 mm for the size at which aggregates start to cause expansion, so it may be that the observed expansion in the 0-2 mm setup is exclusively due to the 1-2 mm sub-fraction. The largest expansion comes from the 4-8 mm fraction, as reported by French [81]. As the mineralogical makeup of all phases is identical, the cause of the larger expansions from this fraction must be due to mechanical causes. A possible explanation is that aggregates of that size neither crack very early as the smaller ones nor very late as the larger ones. Thus the expansion from the gel produced by the reaction of these is neither "dissipated" in paste pores nor confined in the aggregates.

3.3.3 Direction Of Casting

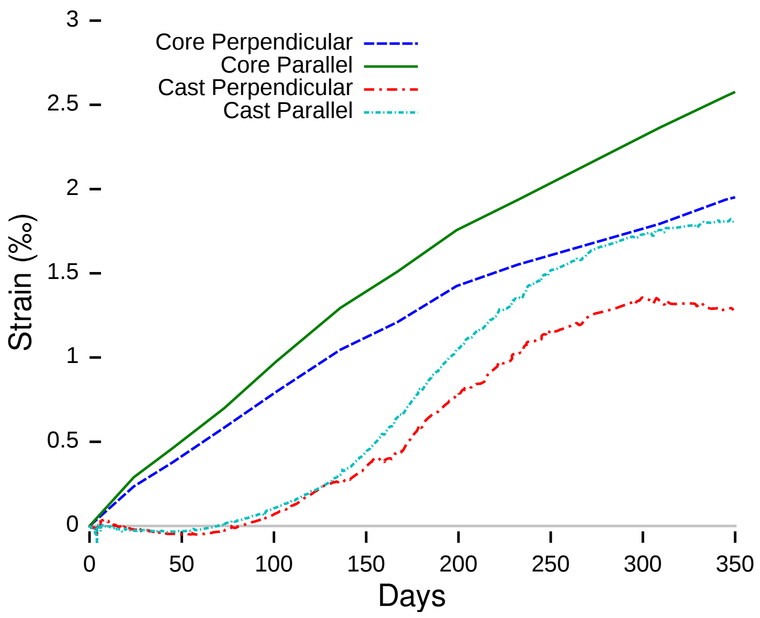
When studying the free expansion of samples due to the ASR, the reaction is assumed to be isotropic. Although concrete is frequently considered at the macro scale to be an isotropic material, it is known to be anisotropic with respect to the direction of casting [90]. Notably, fracture behaviour of concrete at the macro-scale is known to be anisotropic.

The results reported here come from two sets of experiments where expansions were measured in directions parallel and perpendicular to the direction of casting. The expansion of the "cylinders" refers to the expansion of instrumented samples placed in the creep frame but unloaded, and the expansion of the "cores" comes from samples drilled from a single cube of concrete. The cores are measured after a cool-down period, therefore the expansion of the cylinders was corrected for thermal dilation to allow comparison.

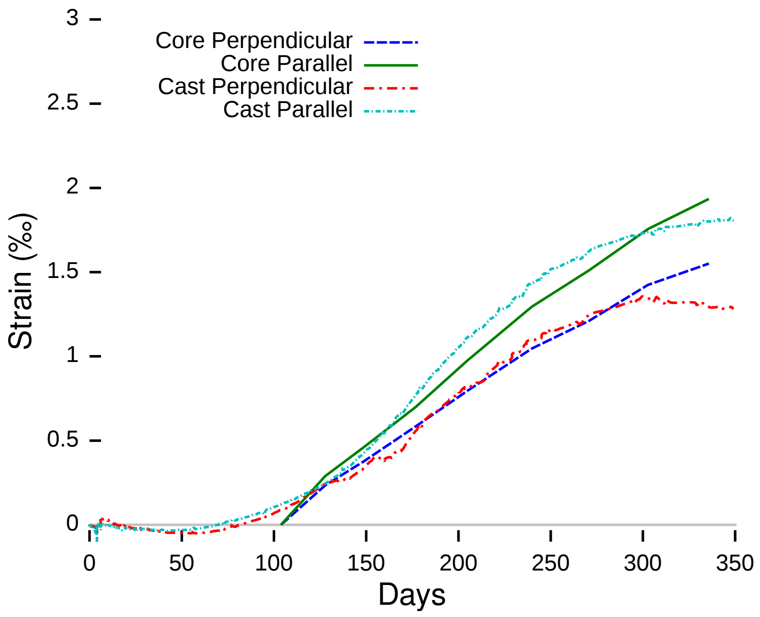
The kinetics of the free expansion of the cores and of the cylinders are different. There is an induction period before expansions accelerate in the cylinders which is not observed in the cores. However, the extent of the anisotropy, measured as the difference of expansion between lateral and longitudinal expansion, is identical[7]. Indeed, when corrected so the initiations of the faster expansion coincide, the match is within the error (Fig. 3.17). The shift is on the time axis for the expansions of the cores, because they exhibit no initiation period. A slight shift was applied to the expansions recorded for the cylinders to compensate for the slight shrinkage measured.

The absolute difference of the lateral and longitudinal expansions between the cylinders and the cores has the same evolution (Fig. 3.18). The values differ only when events we interpreted as sensor slips occur. The difference between the expansions grows linearly with time. As the expansions decelerate with time, the anisotropy is therefore increasing as the reaction develops.

[7]Anisotropy is usually reported in literature as the ratio of the expansions. However, as the sign of the expansions varies during the experiment, such a measure is inadequate for the comparison.



(a) Original values (free expansion)



(b) Shifted values (free expansion)

Figure 3.17: Compared free expansion of the cores drilled in the directions parallel and normal to the casting direction and the free expansion in the transverse and longitudinal directions of cylinders with embedded sensors. The shift is on the time axis for the expansions of the cores, because they exhibit no initiation period. A slight shift was applied to the expansions recorded for the cylinders to compensate for the slight shrinkage measured.

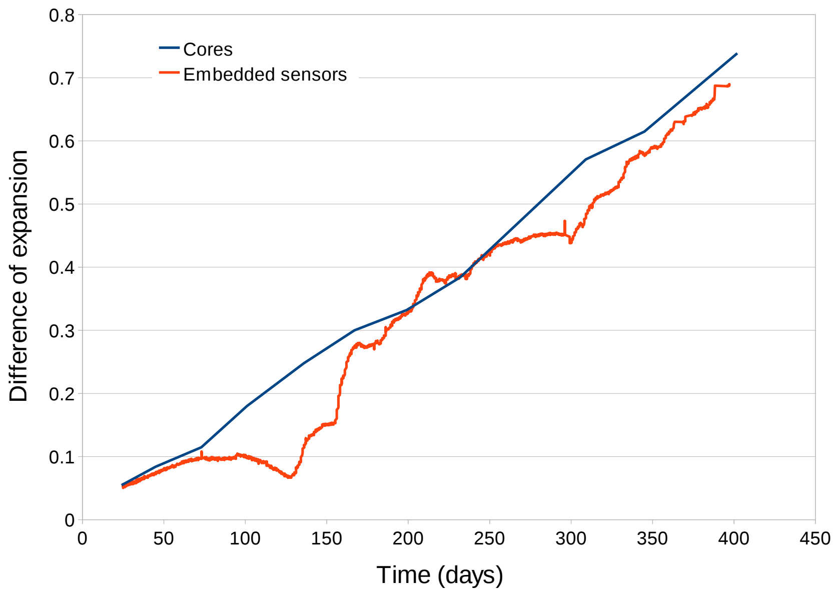


Figure 3.18: Difference in lateral and longitudinal expansions, comparison between cores and the unloaded sample with embedded sensors.

3.3.4 Effect Of Restraining Stress

The ASR induces degradation at the level of the microstructure and expansion. As there is no preferential direction of the silica to react, the anisotropy must be due to an intrinsic property of the microstructure, linked to the placing process.

In real structures, the reaction occurs in concretes which are loaded. Previous studies have indicated that while expansion my be limited in the direction of the restraint, the volume expansion may be redistributed in the unloaded direction. A precise assessment of the redistribution as a function of the stress is required to determine whether this redistribution is simply a mechanical effect or whether there are interactions with the development of the reaction.

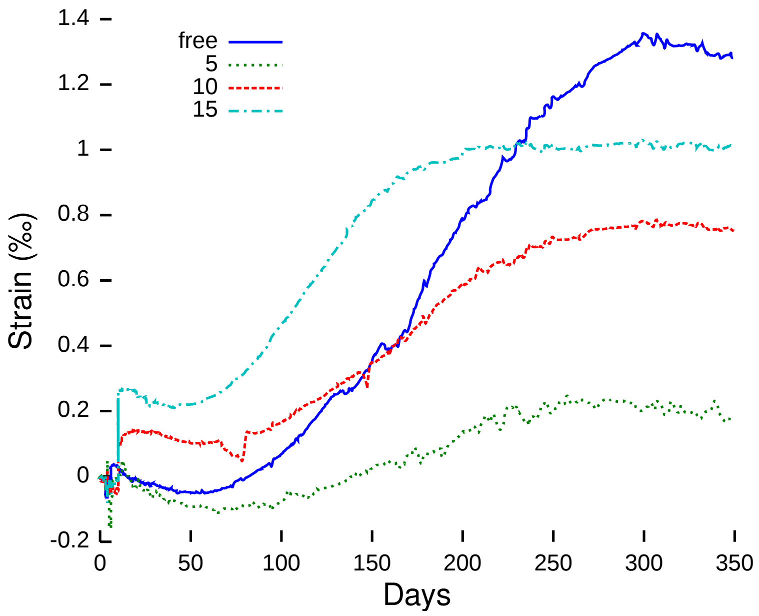
The expansion measured by the sensors for samples cured under various uniaxial loads are reported in Fig. 3.19. These values have been corrected for thermal expansion by calibrating the expansion coefficient using the reported variation in strain of the sensors between 20 °C and 38 °C before the loads were applied, as the hot room was first heated to reach 38 °C, and then the samples were loaded. The lateral expansions increase with the load but remain lower than those observed under unloaded conditions. No visible expansions are observed in the direction of the load.

The expansions for the 10 MPa load apparently show two sensor slip events (Fig. 3.20). The sensor slips are not instantaneous shifts in the recorded expansions, they can last up to 10 days. This indicates that the bonding between sensors and surrounding paste is generally good despite these defects. To verify this, the samples were split using a Brazilian test. The sensors were found well bonded to the cement matrix.

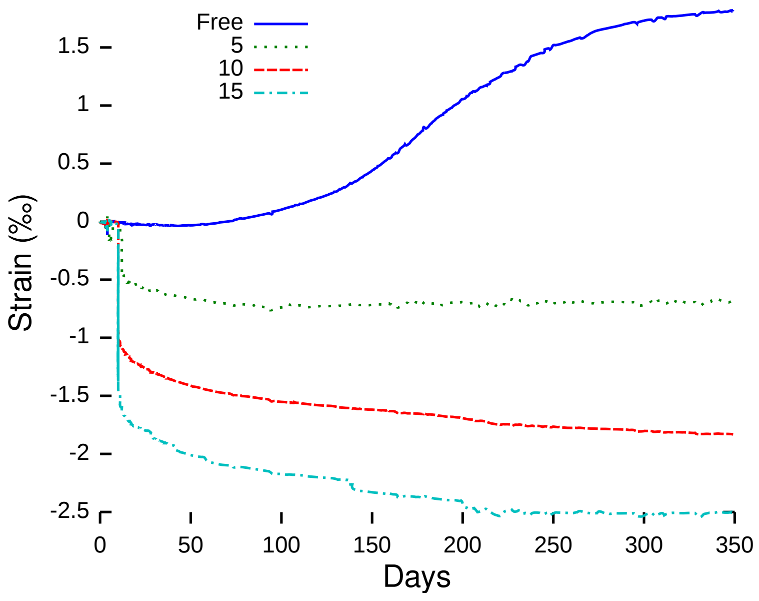
The sensor slip effect was corrected by assuming that the slope of the expansion was the same before and after the event occurred. The expansion results corrected for the slips show that the expansions of the loaded samples, both lateral and longitudinal, are lower than those observed under unloaded conditions (Fig. 3.19). Strikingly, the shape of the expansion curve is markedly different, indicating the damage-expansion process of ASR is affected by the load. To better deconvolute the effects, we have applied a series of corrections to the expansion curves:

- The creep has been removed, using the non-reactive controls as a base
- The elastic component of the deformation due to the loading has been removed.

Under the loading conditions used the samples exhibit significant creep. Non-reactive samples recorded the creep. The deformation due to it was subtracted, by making the assumption that the load induced by the ASR did not affect significantly the creeping behaviour of the samples. Fig. 3.21 shows the deformation of the samples with and without creep. With the creep removed, no longitudinal expansions remain other than due to the elastic deformation of the samples for loads above 5 MPa.

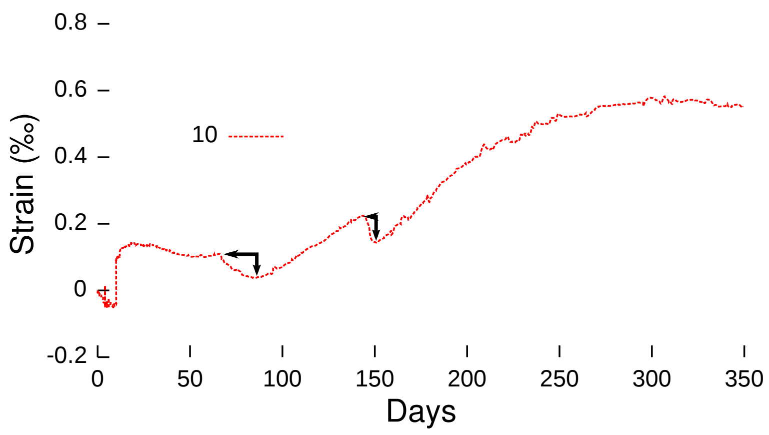


(a) Lateral expansions

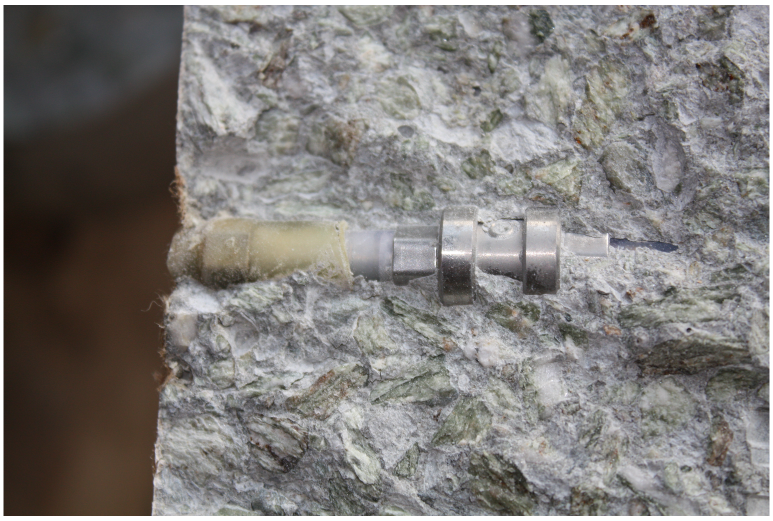


(b) Longitudinal expansions

Figure 3.19: Compared expansions of samples cured under various uniaxial loads. The lateral expansions increase with the load but remain lower than these observed under unloaded conditions. No visible expansions are observed in the direction of the load.



(a) Expansion of an ASR reactive sample under 5 MPa load



(b) Photograph of the sensor after the sample has been split in a Brazilian test.

Figure 3.20: Sensor Slips in the recorded expansion of a sample under 10 MPa load (a). Sensor after a sample was split with a Brazilian test (b). The paste is still attached to the apparatus, which indicates good bonding.

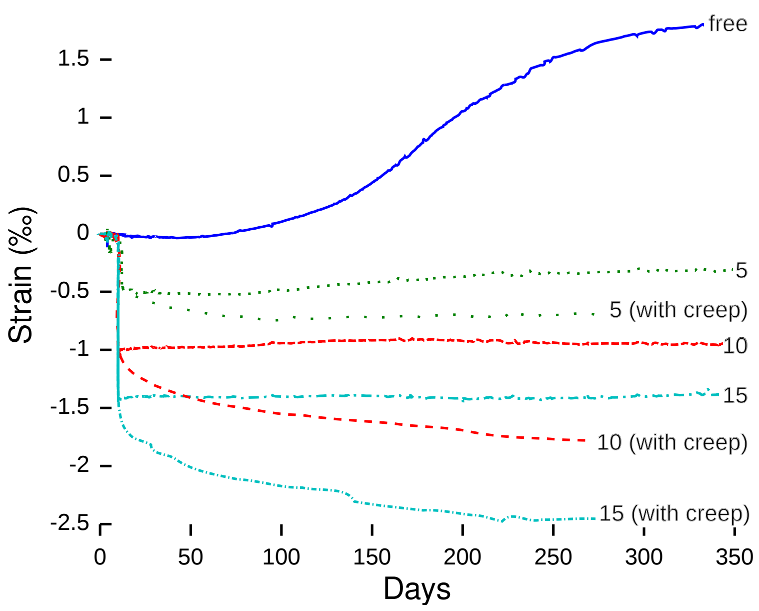
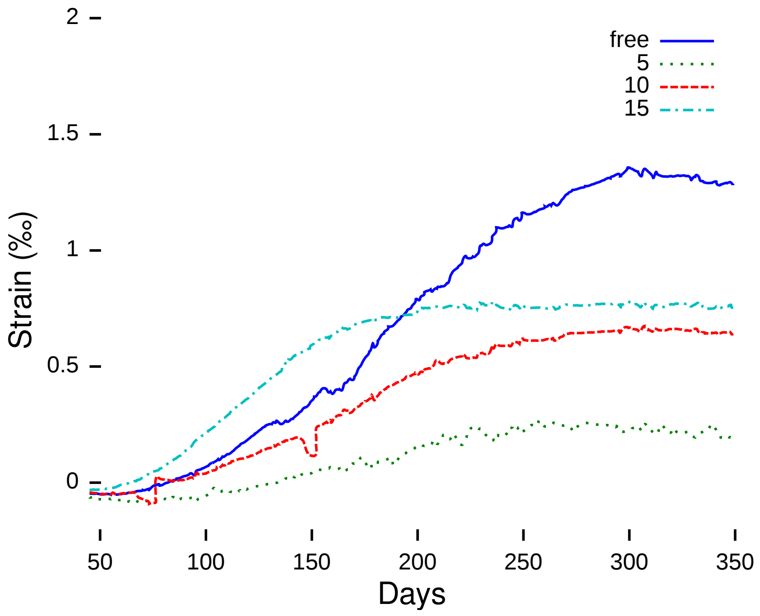
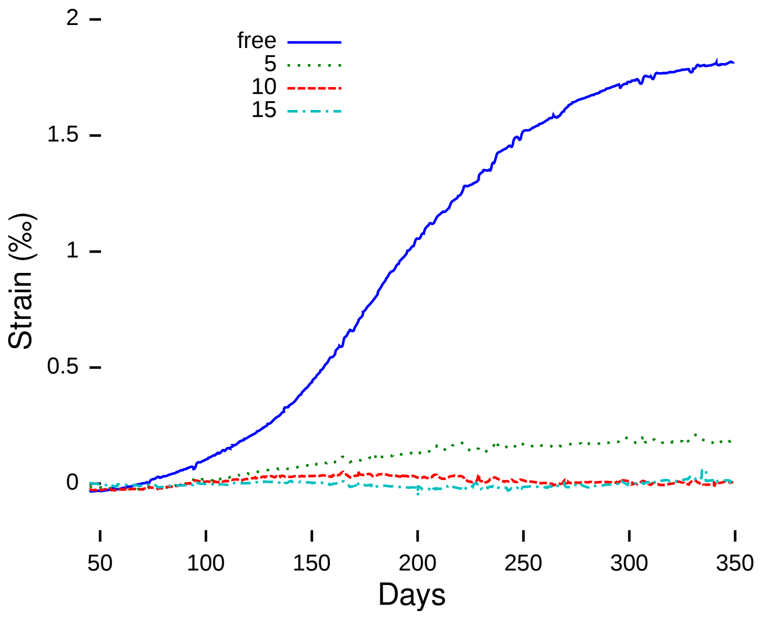


Figure 3.21: Effect of the creep on the expansions. The creep was removed using a measure from non-reactive samples cast with the same cement.

The effect of the initial elastic deformation imposed by the load was subtracted and only the deformation due to ASR is plotted in Fig. 3.22. Eliminating all the non-ASR effects allowed a direct assessment of the effect of stress on the development of the mechanical consequences of the reaction. This analysis shows that of the loading conditions tested, only the free and the 5 MPa exhibit some longitudinal expansion due to ASR. The lateral expansion curve is also affected by increasing the load beyond 5 MPa, notably the expansion becomes larger. Since this effect is purely a consequence of the ASR, and ASR-induced expansion is linked to the damage state of the material, this can be interpreted as increased damage at the microstructure level due to the combined effect of the applied load and the reaction.



(a) Lateral expansions



(b) Longitudinal expansions

Figure 3.22: Lateral and longitudinal expansions of samples due to ASR. Both creep and elastic deformations have been removed. Only the sample under 5 MPa load exhibits any significant longitudinal expansion.

These results confirms the literature reports stating that at a load between 5 and 10 MPa the expansion in the direction of the load is completely inhibited. Another effect is visible: above 5 MPa, the lateral expansion increases with the load and occurs faster. Further analysis indicates that the lateral expansion for the 5 MPa load follows the same kinetics as the free expansion but with $\frac{1}{10}^{\text{th}}$ the magnitude (Fig. 3.23), which indicates that the full transition between expansion regimes is not complete. At 15 MPa, the regime of expansion is completely different. Therefore it seems that increased loads not only increase the damage due to ASR but also its nature.

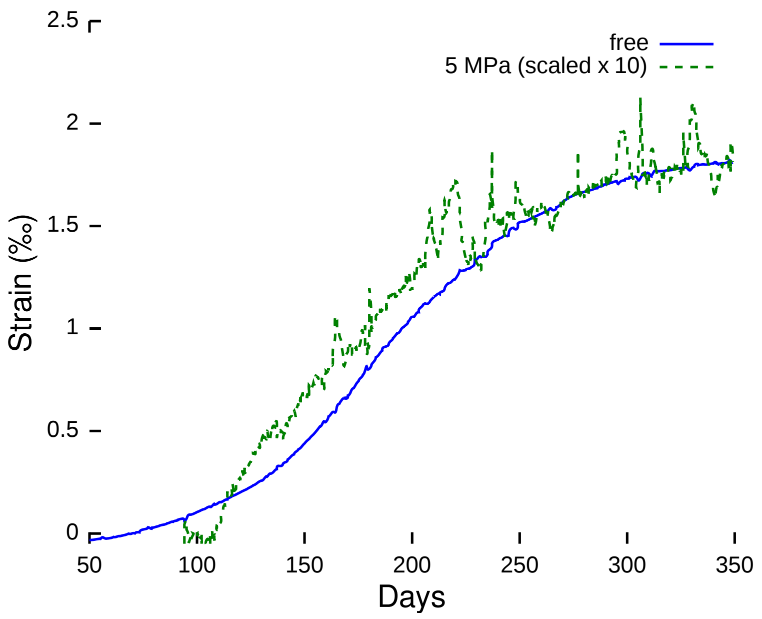


Figure 3.23: Comparison of the free lateral expansion and of the lateral expansion under a 5 MPa load. The 5 MPa expansion has been scaled to highlight the similarity in the curve shape.

Unlike the results reported by Larive [12], we find that the expansion is not simply redistributed. There is a difference of nature in the expansions under unloaded and loaded conditions. Under loads greater than 5 MPa increasing the load increases the kinetics of expansion, whereas below this limit increasing the load reduces the magnitude with similar expansion kinetics. A simple analysis was done where the linear part of the expansion was fitted with a line and the inflexion point marked. This shows clearly the acceleration of the damage under increased loads beyond 5 MPa (Fig. 3.24).

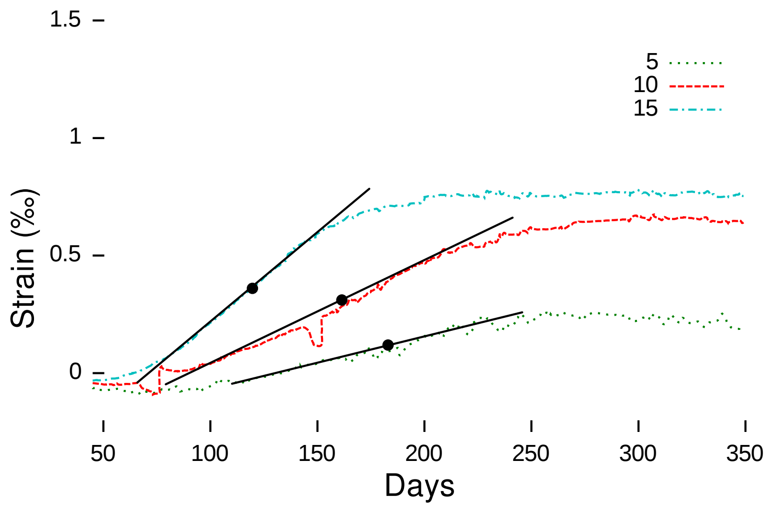


Figure 3.24: Linear part of the lateral expansion of loaded samples. The position of the inflexion point is marked with a black dot. The expansion is accelerated by the load.

The lateral expansions measured after the samples have stopped expanding are reported in Tab. 3.9 and plotted in Fig. 3.25. This allows the comparison of the relative effects of ASR versus the combined effects of ASR, elastic deformation and creep. The most significant factor in lateral expansion is ASR whereas it only plays a minor role in the longitudinal deformations. Indeed, the effect of ASR on longitudinal expansion is nil for the 10 and 15 MPa cases. whereas the elasic deformation only accounts for a small part of the lateral expansion.

Table 3.9: Final expansions recorded as a function of the load.

Load (MPa)	Lat. all (‰)	ASR Lat. (‰)	Long. all (‰)	ASR Long. (‰)
0	1.6	1.6	1.6	1.6
5	0	0	−0.4	0.1
10	0.55	0.7	−1	0
15	0.7	0.9	−1.4	0

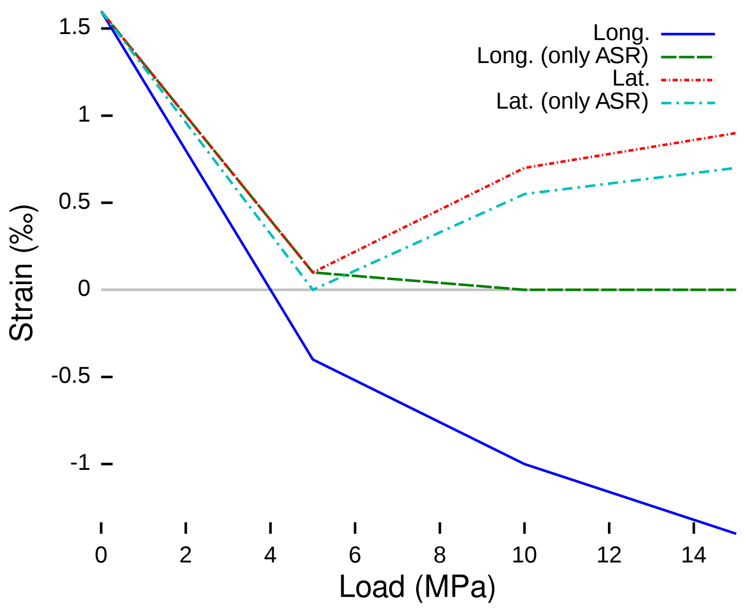


Figure 3.25: Final expansions recorded as a function of the load.

The volumetric expansion of concrete due to ASR should be constant if the effect of stress is the redistribution of the loads due to the reaction. From the measured values, the evolution of the volume of the samples could also be followed. As the sensors gave the strain in two directions, the relative volume (R_V) of the cylinder could be computed as:

$$R_V = \frac{(l_0 + \Delta l) \times (r_0 + \Delta r)^2}{l_0 \times r_0^2} \tag{3.2}$$

With l, l_0 the longitudinal sensor length, respectively initial length, r, r_0 the lateral sensor length, respectively initial length. The results are plotted on Fig. 3.26. This method gives the ratio of the volume after the reaction has occurred over the initial volume. It assumes that the expansions in all the lateral directions are the same. This assumption is reasonable as the only anisotropy of the reaction reported in the literature is with respect to the casting direction, which was normal to the plane of measure considered.

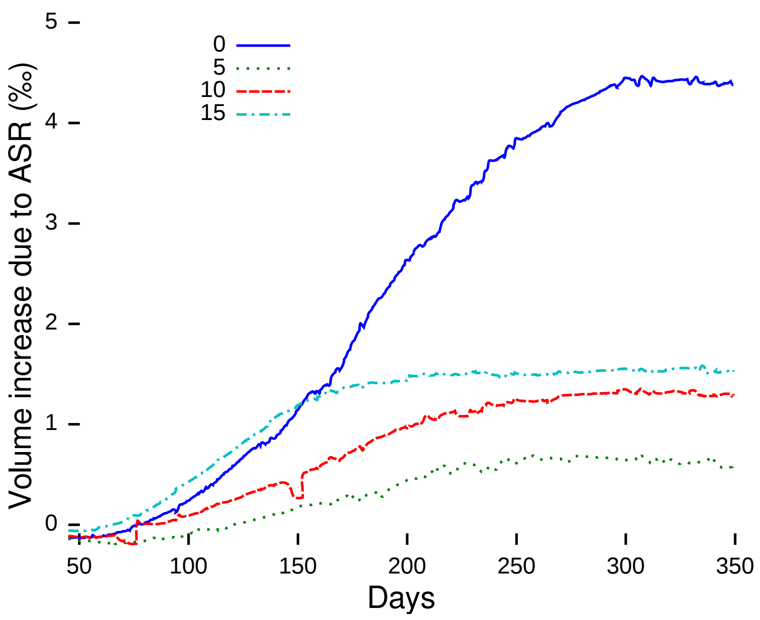


Figure 3.26: Relative volume evolution of the reactive samples under different loads. Two regimes are apparent, one between 0 and 5 MPa, which does not inhibit the apparent reaction, and one between 10 and 15 MPa, which does.

The volumetric variations under 10 MPa and 15 MPa loads are only due to lateral expansion as no longitudinal expansion is observed. Therefore, the volumetric evolution curve is practically the lateral expansion curve squared. Under 5 MPa an increase in load results in a decrease in volumetric expansion. As expansion occurs in only one direction, the differences in volumetric evolution must be due to a change in the mode of damage evolution of the material.

3.4 Conclusion

The recorded expansion of ASR was found to depend on the mix design parameters as well as the external loads. The effects observed show no general trend which could be fitted using simple regression techniques. Therefore more powerful modelling techniques are required to understand the observations and link the microstructural effects of the ASR to the macroscopic observations described in this chapter. The next chapter describes the development of a numerical tool suitable to run the necessary simulations.

CHAPTER 4

Numerical Modelling of the Alkali-Silica Reaction Degradation Mechanism

Parts of this chapter have been published in Cement and Concrete Research.

4.1 Introduction

The alkali-silica reaction (ASR) is characterised by the breakdown of silanol bonds in poorly crystallised silica found in aggregates in the presence of alkaline ions. The product of this reaction is an amorphous alkali-silica "gel" which expands in the presence of moisture, inducing stress in the microstructure. This causes concrete expansion and mechanical properties degradation. Prediction of the free expansion, as well as the subsequent degradation of the stiffness and strength are of great importance for the owners and managers of affected structures.

Mechanical modelling of ASR at the material level poses significant challenges both technical and scientific. A model based on physics is necessary to understand the degradation mechanism and to permit prognosis and life-time assessment of affected structures. Such a model should be based on a detailed meso-mechanical representation of the concrete microstructure to capture the variety of mineralogies found in the field.

In this chapter, we present a micro-mechanical model of ASR implemented using a purpose-developed extended finite element modeling (XFEM) framework capable of a full scale simulation of laboratory-sized samples in two dimensions. This framework called Automated meshing for integrated experiments (AMIE) was developed to be very flexible to enable study of various proposed ASR mechanisms at the meso-structural level[41, 78].

4.2 ASR models in literature

ASR models are frequently formulated at the material level for use in structure-level codes, and use empirical relationships [95, 15, 105, 8]. Constructing models from semi-empirical relations requires extensive experimental campaigns such as in the theses of Poyet[15] and Larive [12], where sufficiently large datasets are produced which allow robust fitting. The curing conditions, temperature, stress, and relative humidity are varied, and the resulting evolution of expansion over time is measured as function of those parameters. The findings from such approaches are extensive and detailed, but are specific to each aggregate type. Coupling is often introduced between various parameters, even when these have no direct physical connection, such as aggregate size and alkali content. Another approach consists in the development of analytical models. Relationships predicting the damage at the material level are derived from assumptions on the mode of reaction of the aggregates. These models mostly assume gel formation to occur preferentially at the interface between aggregate and paste [106, 87]. The degradation of mechanical properties is described as coming from the ASR gel which first fills the pores and then induces expansion and cracks.

Numerical models at the level of the microstructure are rare. For example, such models focus on the fracture mode of a single aggregate such as in the work of Çopuroğlu and colleagues [107], which computes crack propagation in a single aggregate using a lattice model.

Models can be formulated from phenomenological or mechanistic points of views. The lack of consensus about the origin of the degradation mechanism on one hand, and the difficulty of measuring the advance of the reaction rather than its consequences on the other hand make the formulation of experimentally based mechanistic models arduous. Nevertheless, there are some publications which relate macroscopic consequences to a measure of the reaction. Garcia and colleagues studied the evolution of the pore volume inside aggregates [108],and measured the evolution of the proportion of the various types of silica tetrahedra during the swelling. They propose a mechanism based on a diffuse reaction in the aggregates causing microcracking, and note that the relation between expansion and reaction is strongly influenced by the degradation state of the aggregates and paste. Ben Haha and colleagues linked the fraction of the aggregates which reacted to form ASR gel to macroscopic free expansion.[1].

Whether the models be analytical, semi-empirical or numeric, they rely on underlying hypotheses about the degradation mechanism. The manifestation of the reaction has been shown by Ponce and colleagues to depend on the mineralogical nature of the aggregates [7]. Aggregates such as opal or vitreous volcanic rocks react from the surface, with a predominance of gel formation at the paste-aggregate interface. Mixed mineralogy aggregates, which are more common in the field, have reactive sites dispersed throughout their volume. A numerical framework capable of integrating the various microstructural manifestations of the reaction is therefore necessary to model ASR in general.

4.3 Model

The microstructural model presented here is based on the AMIE framework, which integrates a set of tools to provide the necessary components for the simulation of concrete at the microscale: geometry library, mesher, finite and extended finite element libraries [109], solvers, post-processors. It is optimised to integrate a number of enrichment sources much larger than that commonly available in other software [47, 41].

A typical simulation of ASR with AMIE is as follows. From a particle size distribution, aggregates are generated and placed in a sample. In each aggregate reactive zones are generated and placed. The framework then generates the discrete representation of the setup (Fig. 4.1). At each step, boundary conditions are applied, and the reactive zones are caused to expand. The damage caused by this expansion is computed, and the macroscopic properties are extracted. The process is then repeated a specified number of steps. To investigate in detail the effects of ASR at the meso-scale, an explicit numerical representation of many factors is required. This implies large simulations, namely a finely meshed standard mortar bar cross-section of 40 mm $\times$ 40 mm. Such large simulations are necessary as crack-paths within aggregates spanning the entire particle size distribution will be produced; direct comparison of the simulations with experimental data is direct and furthermore features such as the wall effect at mould surface are also simulated.

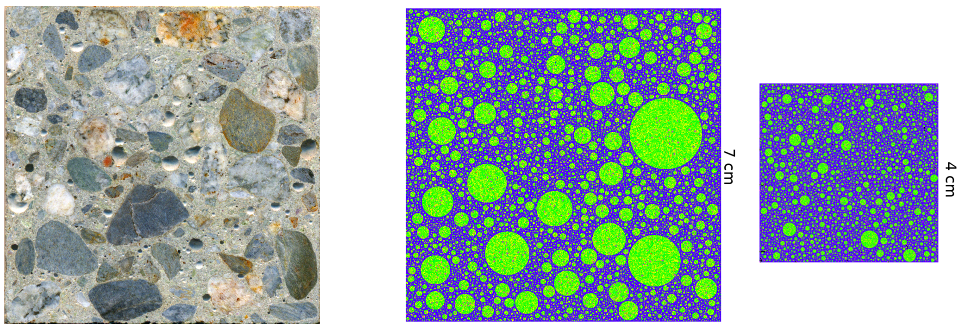


Figure 4.1: Simulated concrete (centre) and mortar microstructures (right), compared to an actual slice (left)

Spherical aggregates are used for simplicity. Indeed, the particle size distribution (PSD) affects the expansion, whereas no particular effect of shape has been identified. Also, the aggregate shape will have little effect because the reactive zones grow inside the aggregates. The microstructure is generated using a random packing algorithm, which allows a packing density of 63% of volume with aggregates to be achieved in a few seconds. The particle size distributions of the aggregates is taken from that of the real mortars and corrected for the three dimension to two dimension slicing effect. The ratio of largest to smallest aggregate is 50, which is a cut off at 200 μm.

The reactive zones are explicitly positioned within the aggregates. As these reactive zones are much smaller than the aggregates and growing as the reaction proceeds, meshing them would yield problems too large to solve in reasonable time. Therefore an XFEM representation for the growing reactive zones was chosen to allow their explicit representation, position, size and evolution. Although the framework is constructed to allow simulation in three dimensions it is not practical at present to manage the large amount of data required for such simulations at this resolution, therefore the current approach is restricted to two dimensions.

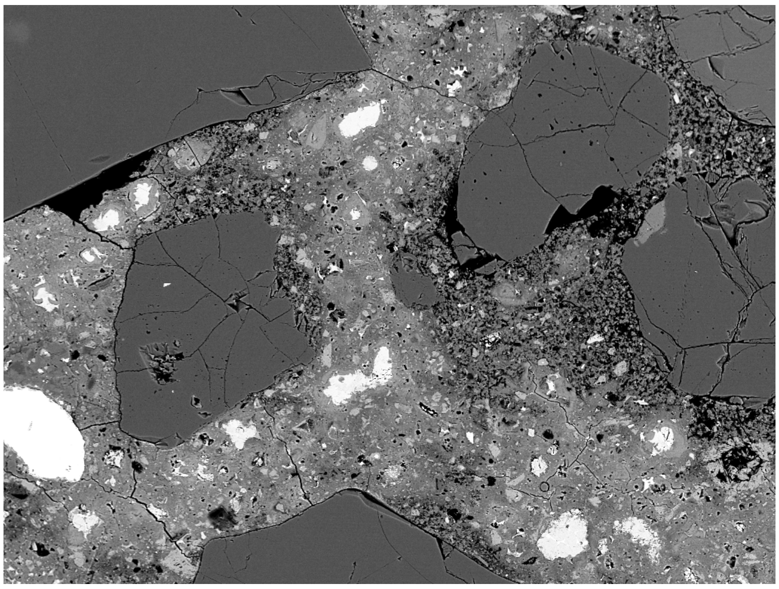


Figure 4.2: Micrograph of a sample after a ten weeks cure immersed at 40 °C. The cracks initiating at the zones within the aggregates which reacted are clearly visible.

The mechanical properties of paste and aggregates (stiffness, critical stress) were obtained experimentally. The simulation of the intrinsic variability of the paste and aggregates is important to stabilise crack propagation, as well simulate crack initiation at local defects. Aggregates in ASR-affected structures exhibit variable mineralogies and so their mechanical properties will vary at the local level. The influence of the varied nature of the aggregate is simulated by varying randomly the distribution of the mechanical properties of the elements making up the aggregates. This local variability is modeled by having the mechanical properties $\mathcal{P}$ of the elements follow a simple statistical law:

$$\mathcal{P} = \mathcal{P}_{\text{Prescribed}} \cdot (1 - \xi) + \mathcal{P}_{\text{Prescribed}} \cdot \xi \cdot \omega \tag{4.1}$$

In this equation, ω is a random Weibull variable and the the randomised fraction of the property, ξ, is set at 0.2; this fraction follows a Weibull law of mode $\mathcal{P}_{\text{Prescribed}}$. This factor was set to reproduce the experimental spread observed in mechanical tests.

The paste fills the space not occupied by the aggregates and has similarly randomised properties. Both aggregates and paste are assumed to be linear elastic, with damage.

Once the microstructure has been generated, it is meshed conformantly using a Delaunay mesher, developed in-house around the core meshing algorithm from Devilier and colleagues [52]. The samples were finely meshed to capture the fracture pattern within the aggregates and the paste.

4.3.1 Damage Model

Damage at the meso-scale is induced by a dense network of cracks. Because of the multiplicity of those cracks, special care must be taken that their propagation corresponds to a global energy minimum. The algorithm outlined in Fig. 4.4 ensures this is the case, even if multiple fracture criteria and damage mechanisms coexist. This approach leads to simulated fracture patterns which are qualitatively similar to the experimentally observed ones (Fig. 4.3).

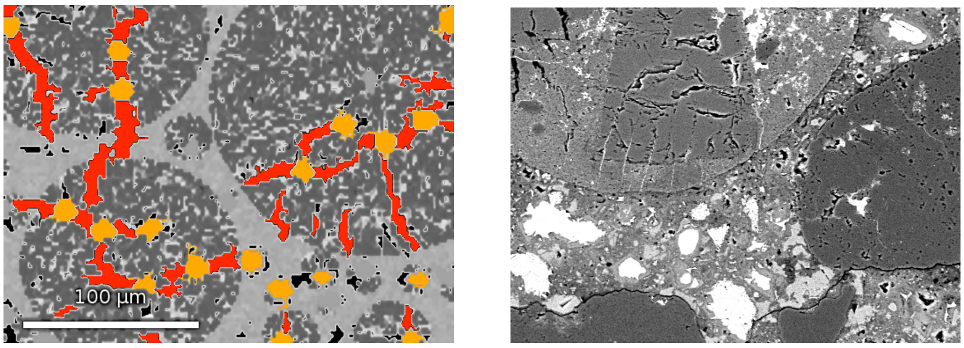


Figure 4.3: Detail of a fracture pattern generated by the ASR model (left). For comparison, a micrograph at the same scale of an aggregate (right).

Also, the more usual method of iterating on the load state to match the damage is not applicable, as the damage is induced by a load prescribed by the setup of the simulation. Thus, the fracture propagation to the point of equilibrium must be determined *a posteriori*, after the load has been set. As both paste and aggregates can be considered quasi-brittle, we have opted for the following damage evolution law where the damage evolution parameter η is adjusted to the experimentally measured material parameters and ε is an arbitrarily small value.

$$d^{i+1} = \max(d^i + e^{\eta d^i}, 1 - \varepsilon) \tag{4.2}$$

Where d is a factor such that the Young's modulus E is related to the original modulus by the equation 4.3.

$$E = (1 - d)E_0 \tag{4.3}$$

The damage is incremented when a failure criterion is reached. The failure criterion used in our simulations is a modified Mohr-Coulomb criterion. It is reached when the maximum principal stress is beyond a critical value in tension or compression.

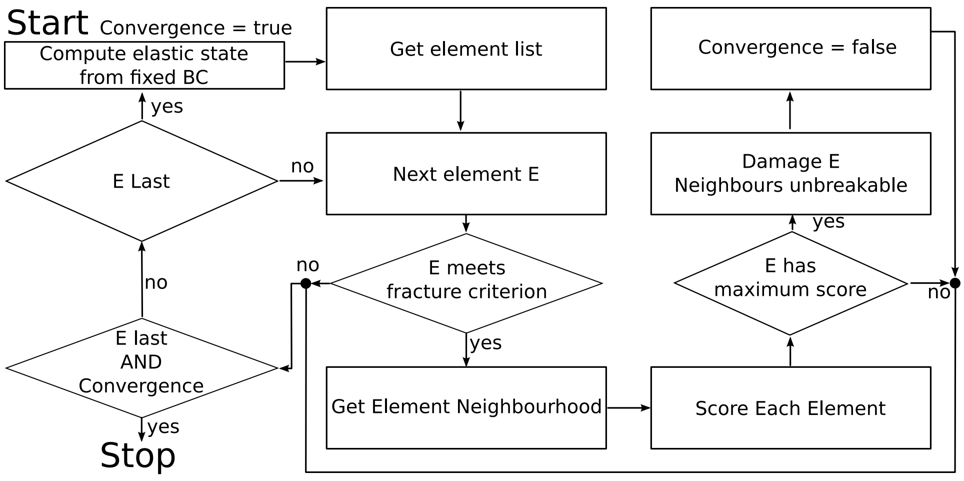


Figure 4.4: Flow diagram of the algorithm which determines which element must be damaged. This method ensures that at each step of the cracking process the damage increment minimises the global energy. If the increment is small enough, this method effectively reproduces the damage history which corresponds to the global energy minimum.

To ensure that the damage is applied on a non-local basis, a neighbourhood is defined for each element which defines the density of elements which can be damaged at each step of the damage computation. A neighbourhood of the size of the average inter-aggregate distance yielded results closest to the experiments. To determine which elements should be damaged at each step of the damage computation loop, we defined a score which allows inter-element comparison. This score is based on the distance to the fracture surface defined by the criterion. In those simulations, the score s is calculated as defined in equation 4.4.

$$s = 1 - \frac{\sigma_{\max}}{\sigma_{\text{crit}}} \tag{4.4}$$

4.3.2 Gel Model

Microscopic observation of a wide range of mineralogically different aggregates consistently revealed the presence of ASR gel pockets in the aggregate. The originality of our model lies in the fact that the gel pockets are explicitly considered, so the damage in the aggregates and paste emerges from the numerical setup of the meso-structure directly. The implementation of this feature required the use of modern developments in both numerical theory and numerical methods.

Gel pockets are modelled using a soft-discontinuity type of enrichment. The integration step is performed by generating a temporary conformant tessellation of each enriched element. The generated mesh is then refined until convergence of the domain integral of the enrichment functions is reached. The final quadrature generated in this way is then used for the integration of the weak form defining the behaviour, and the temporary mesh is discarded (see In App. C.3.2 for more details).

Using this approach, we can model the gel by taking into account the spatial distribution of the reactive zones, their size and the interface between the gel and the aggregates. The gel properties are assumed to be linear elastic with an imposed strain, as the gel is constrained by its surroundings until large amounts of damage have occurred. Using XFEM, we can accurately represent the geometry in each simulation step as illustrated on Fig. 4.5.

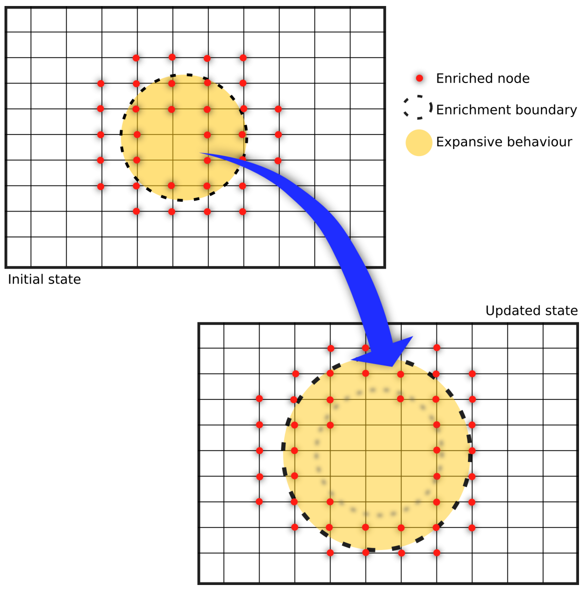


Figure 4.5: Update of the enrichment scheme. The scheme is illustrated on a structured rectangular mesh for clarity, however, the meshes used in the simulations are triangular unstructured.

The gel mechanical properties are not well known, and experimental measures exhibit important variability in terms of the chemical nature of the gel[11] and its apparent mechanical properties[9, 110, 10]. However, the chemistry of the ASR gel, is close to that of calcium silicate hydrate (C-S-H). For this reason, we have decided to assume that the gel is quasi-incompressible, with a Poisson ratio of 0.49997, that of water, and a stiffness which is a fraction of that of C-S-H [111], expressed in the figures as α.

The free expansion of the gel is, like the mechanical properties, difficult to measure experimentally. However, it can be obtained by fitting the early part of the free expansion-damage curve. In the very early stages of the reaction, the damage from induced cracks is very little, and the expansion of the sample can be assumed to be elastic. From this we can obtain the expansive property of the gel which would result in this expansion. We found that a 50% volume expansion fits the experiments. The variability of the image analysis method at the early age of the reaction makes it difficult to be very precise about this value. However, the overall result of the simulation is not very sensitive to the value taken and 50% is also consistent with the stoichiometric volumetric ratio reported by Garcia and colleagues [108]. The strain of the gel, ε_{imp} is imposed as a virtual force on the nodes of the elements inside or cut by the gel pockets. The integration scheme is generated such that only the fraction of the element where the swelling takes place is accounted for (Eq. 4.5).

$$\mathbf{f}_i = \int\limits_{\Omega_{\text{gel}} \cap e} \nabla h_i \mathbf{E} \varepsilon_{\text{imp}} \, \mathrm{d}(\Omega_{\text{gel}} \cap e) \tag{4.5}$$

Ω_{gel}, the gel surface, e the element surface, h_i the shape function associated to the degree of freedom considered, $\mathbf{f}_i$ the associated force, $\mathbf{E}$, the Cauchy-Green stress tensor of the gel.

4.3.3 Reaction Mechanism Model

The model presented here is at present achronic. The sample expansion is computed only against the degree of reaction, which is the part of aggregate which has reacted to form ASR gel. Thus the mechanical model cannot yet take into account such time-dependent behaviour as gel flow and paste creep depending on the curing conditions. As reaction could take weeks or years, creep may be important and we expect our model to predict expansions higher than the experimentally measured ones at the later stages of the reaction, as no relaxation mechanism is present other than cracking.

The gel is grown by steps in each aggregate until a preset percentage (3%) of the considered aggregate has reacted, at which point we stop the reaction in that aggregate. This value was calibrated from the results of Ben Haha, who had observed an exhaustion of the expansion at that level of reaction. To verify the influence of gel growth capping per particle, we ran simulations where all gel pockets grow in the same way, independent of their location. The gel is principally located in the larger aggregates, as they form the main fraction of aggregate volume. Thus, stopping the reaction in the smaller aggregates has little effect on the final expansion. However, the initial shape of the expansion-reaction curve is only slightly affected.

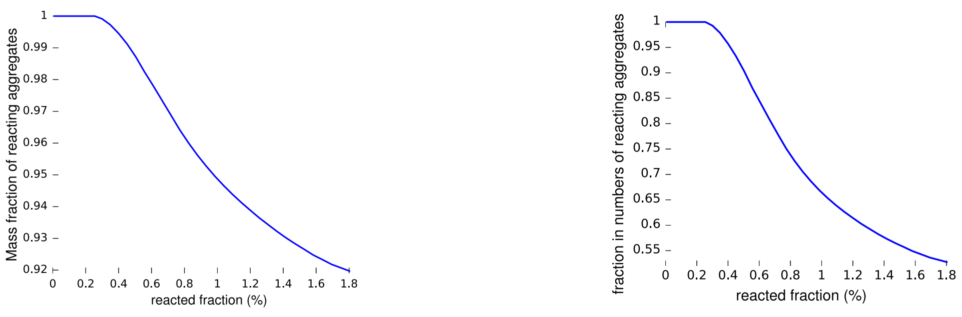


Figure 4.6: Fraction of the mass of individual aggregates where the reaction has been stopped as a function of reaction advancement (left). Fraction of the number of individual aggregates where the reaction has been stopped as a function of reaction advancement (right).

By the end of the run nearly half of all aggregates have exhausted their expansion potential (Fig. 4.6). This effect can be simulated explicitely in our model as the full PSD is represented.

4.4 Results

4.4.1 Simulation of the Free Expansion of Mortars

The simulated expansion-reaction curves follow the early part of the experimental dataset, but reach a plateau at higher expansions. The jitters observed in the simulated curves start at the onset of paste failure, which happens at the same reaction level as in the experiments. The higher final expansion can be explained by the absence of strain relaxation mechanisms in the paste other than crack propagation: there is no creep (Fig. 4.7).

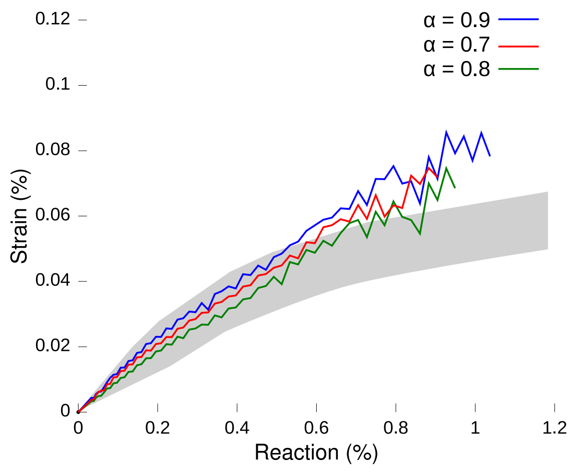


Figure 4.7: Simulated expansion-reaction curve for mortar. The curves are not prolonged because a state is reached where the samples are not of a single piece anymore.

As shown in Fig. 4.8, the measured damage and the actual reaction diverge. For this reason we have only simulated expansions up to 1% of reaction. At this level significant damage has occurred (Fig. 4.9), beyond what would have been considered critical in a structure.

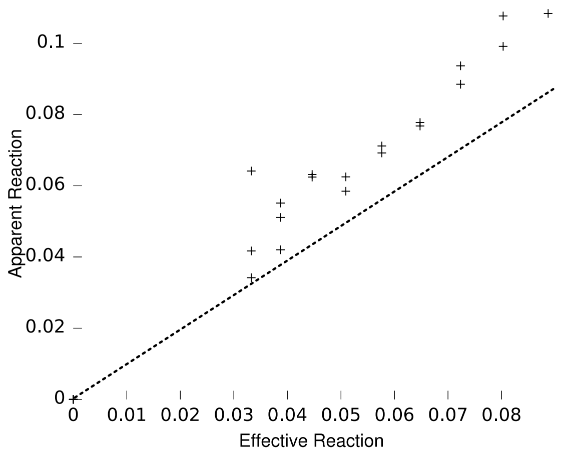


Figure 4.8: Apparent reaction as a function of effective reaction. The dotted line is the line of equality.

The expansion-reaction curves obtained from simulation match the experiment well within the observed variability, with no fitting parameters other than the stiffness and free expansion of the gel. We found that the range of gel stiffness which could fit the experimental relationship is quite large: between 0.6 and 0.9 of that of C-S-H.

4.4.2 Prediction of the loss of mechanical properties

From the average stress and strain of the samples at each time point, we can compute the apparent stiffness of the sample, and thus link the advancement of the reaction to the damage level in the sample. These results are then compared to experimental values reported in the literature (Fig. 4.9).

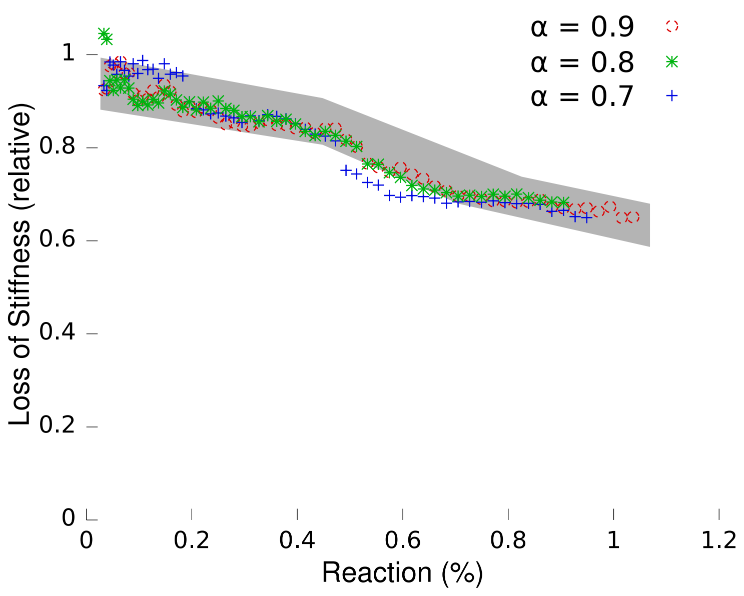


Figure 4.9: Stiffness as a function of the progress of the reaction (left). The loss of stiffness as a function of expansion is compared to a range of reported values from Ben Haha (grey shade). α is the ratio of simulated gel stiffness to C-S-H stiffness.

The loss in stiffness is mostly due to the aggregates cracking, as the paste is mostly in compression, with stress levels below the elastic limit. Paste failure occurs only when cracks from the aggregates reach the paste, thus increasing the tensile stress locally and initiating failure there. This also means that the properties of the interfacial transition zone between paste and aggregate, which have not at present been included in our model will have negligible impact on the damage evolution. The loss of stiffness predicted by the simulation is consistent with data reported in the thesis of Ben Haha[2], but is sensitive to the choice of gel properties at low reaction values.

4.4.3 Sensitivity to the Various Parameters

Several simulation campaigns were run to verify that our results are not sensitive to numerical effects. First we verified sensitivity to mesh size. The characteristic element size was halved, yielding four times as many elements, and the same simulation was run. The results are not significantly different when using a much finer mesh. This verifies both the XFEM model for the expansive zones and the energy-conservation of the damage model. The expansion values vary by only 2% at the final degree of reaction (e.g. at 1% expansion, the error is $\pm$ 0.02%). Another second simulation campaign was run with a microstructure as dense as the packing algorithm allows. A mortar sample was generated with a packing density of 71%, larger than the value from the experimental mix design. The final expansions differed by only a relative amount of 6%, which is consistent with the increase in reactive material.

Finally we also verified the sensitivity to the variability of the local mechanical properties. We varied from 0.2 to 0.5 the weight ω (See Eq. 4.1). Increase in variability lead to earlier cracking, but did not affect measurably the results.

4.5 Discussion

4.5.1 Correlation Between Damage and Reaction

Image analysis cannot distinguish well between reaction (gel or voids) and damage (cracks): both appear as dark zones in the aggregate. In our simulations we compared the apparent reaction, as would be measured by the image analysis and the effective reaction, which is given as an input in the program. As expected, the apparent damage is larger than the effective reaction percentage. However they are close, and quasi-linearly related, if divergent. The cracks do not represent a significantly large volume of damage, compared to the volume of the gel pockets (Fig. 4.8).

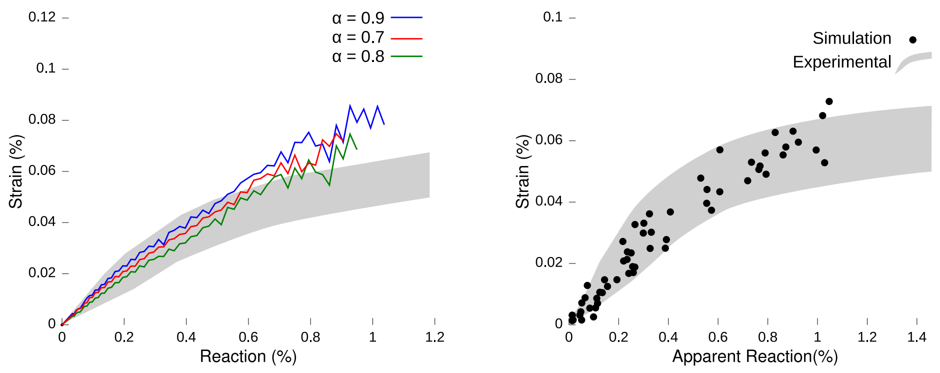


Figure 4.10: Comparison between the real expansion-reaction curve (left) and the apparent one (right). The point where the expansions reach a plateau is higher than the experimentally measured values in both cases. This indicates that a relaxation mechanism is not taken into account in the simulations.

The apparent expansion-reaction curve is less sensitive to the fit parameters than the real one (Fig. 4.10). This is explained by the strong link between damage and expansion, which is more direct than the link between expansion and reaction. This further explains the low variation observed experimentally across aggregate types and curing conditions. The noise apparent in the simulated damage-expansion curve is due to the healing effect of gel growing over fractured aggregate matter. This variation is consistent with experimental variability.

4.5.2 Expansion Mechanism

The expansion-reaction curve exhibits three regimes: linear expansion, aggregate cracking and paste cracking (Fig. 4.11). The simulation shows that this is the result of four mechanisms: the expansion caused by the gel, the relaxation from the damage, the interaction between the gel zones, and the creep. The growth of the reactive zones can be measured as the reaction advances, and provides an insight into the expansion mechanism. In the first stage, as the gel progressively damages the aggregates, the expansion becomes less and less restrained until a plateau is reached, at this point the gel is essentially free to expand in the aggregates. This plateau comes to an end as the paste starts restraining further expansion. This evolution is mostly governed by the evolution of damage, and is not very sensitive to the gel stiffness parameter, but rather to gel localisation. It should be noted that the expansions at the end of the first stage are already well in excess of those likely to cause structural problems in large unreinforced structures such as dams.

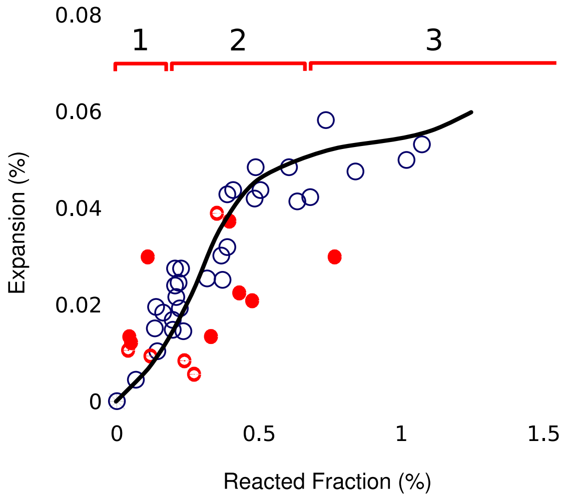


Figure 4.11: Mechanism for expansion and degradation. (1) Elastic expansion; (2) Aggregate failure and gel pressure buildup; (3) Paste cracking.

As expected, the simulated expansion levels off at a higher point than the experimental ones (Fig. 4.10). This can be explained by the time-dependent visco-elastic behaviour of the gel and the creep of the paste not being modelled in these simulations, which leads to less strain relaxation than in real samples.

Modelling observations show that the paste is in compression when all aggregates are expanding (Fig. 4.12). This compression is sufficient to inhibit the expansion which would be caused by the smaller fractions, and the expansion caused by the larger fractions is still low because the aggregates have not yet started to crack significantly. The strain of the gel, as a measure of confinement, is reported on Fig. 4.12. The gel pockets are *more* confined during the initial stages of the reaction, then as the aggregates crack, the compressive strain decreases. Thus, the apparent (positive) strain observed in the specimen is due to the aggregates being broken apart. Identical behaviour was observed for gel stiffness varying between 0.9 and 0.7 times that of C-S-H (this ratio is denoted α in the figures). This indicates that this effect if of geometrical origin.

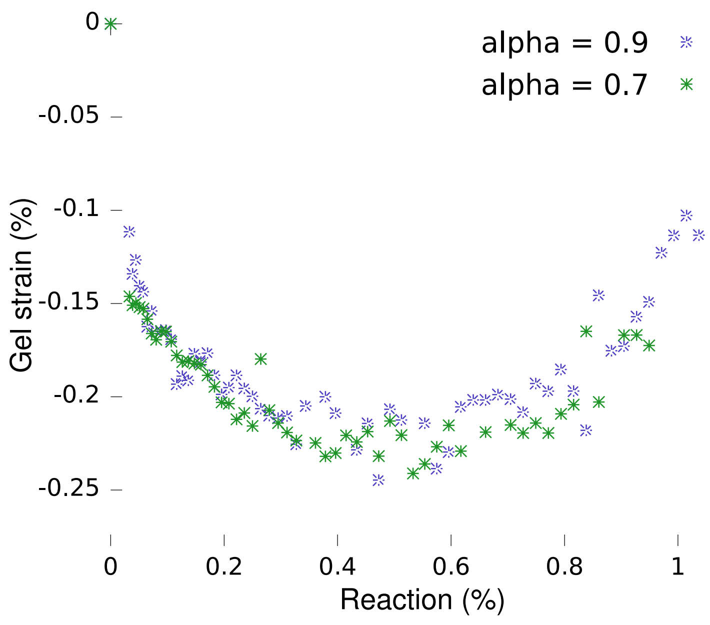


Figure 4.12: Gel strain in a mortar bar as a function of amount of reaction. The strain of the gel pocket becomes more and more negative as the reaction advances, until it goes up again. This indicates that initially in confinement, the gel pockets are *more* confined during the initial stages of the reaction, then as the aggregates crack, the compressive strain decreases. Thus, the apparent (positive) strain observed in the specimen is due to the aggregates being broken apart. The low sensitivity of the observed confinement with respect to the gel stiffness highlights the geometrical origin of this observation.

4.5.3 Comparison to Simpler Models

Various simplifications have been proposed for meso-scale modelling of ASR (Fig. 4.13). The simplest model applies a pseudo-thermal expansion for the aggregates as a source for the expansion. We found it possible to calibrate such model to fit the early part of the curve, using a linear correlation between the imposed strain and the degree of reaction. However, such a model does not capture the damage in the aggregates, and thus the loss in physical properties (Fig. 4.14).

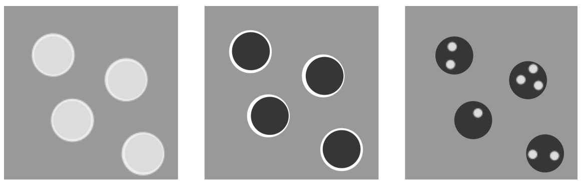


Figure 4.13: Three Models for ASR-induced expansion. Homogenised aggregate expansion (left), gel rim expansion (centre), gel pockets expansion (right).

The predictive nature of such a model is low, as the expansion of the aggregate must be computed as an empirical function of the degree of reaction. Such models have been used in the past to predict free expansion, using reduced particle size distributions. However the simulations demonstrates that such models are extremely dependent on the PSD of the aggregates, and exhibit very different damage patterns depending on the accuracy of the representation of the microstructure. Simplification of the PSD has consequences on the evolution of damage. A dense packing of aggregates causes all the paste to be in compression, whereas a reduced packing allows for higher shear levels in the paste, which then leads to cracking at lower levels of macroscopic expansion in those simulations.

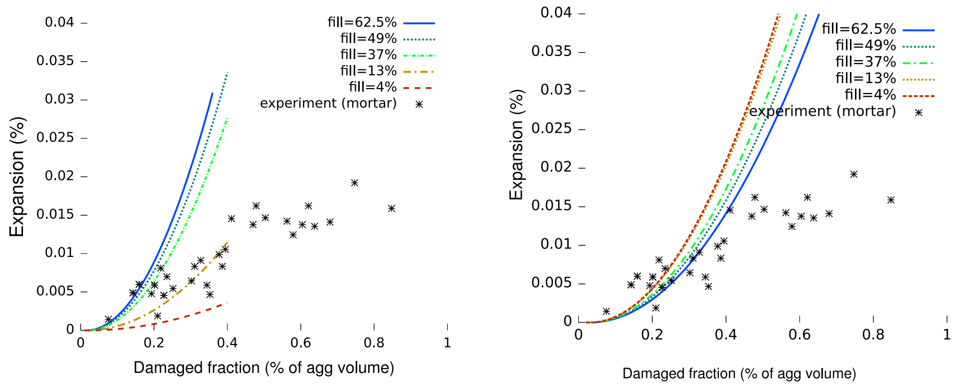


Figure 4.14: Expansions predicted by a simplistic model, with varying cutoff points in the PSD (left). Expansions have been renormalised according to the aggregates content (right).

When the local stress around the aggregates reaches a critical threshold, the sample undergoes critical failure. We found the critical failure to occur at the same imposed expansion, independently of the aggregate content.

The expansion-reaction curve observed experimentally exhibits different regimes: linear expansion, aggregate failure, paste failure(Fig. 4.11). The simplified model can capture the first and last regimes. When the local stress around the aggregates reaches a critical threshold, the sample undergoes catastrophic failure. The stress imposed by the aggregates needs to be fit with an empirical model which would include the effect of the damage in the aggregates. As such, this simplified model is incomplete if using only directly measured experimental values.

Another simple model for the reaction which has been proposed is that the gel is formed as a rim around the aggregate. Such a microstructural makeup has never been explicitly simulated, but this vision of the reaction is used as the source of other semi-empirical models, such as the growing aggregate model described above. This model can be correlated to an advance of the reaction, as the gel localisation and amount are explicitly defined for each step of the reaction. To test this model, we implemented an enrichment scheme which could reproduce the two distinct interfaces between the aggregate and the gel, and between the gel and the paste.

This model is different from the precedent in that it explicitly affects the bond properties between aggregates and paste.

The fracture patterns obtained in such a setup show this model is not a good candidate to explain the ASR degradation: the cracks are located in the paste, while the aggregates remain largely intact, except for cracks initiating at the interface. Considerable decohesion between the aggregates and the paste is also observed (Fig. 4.15).

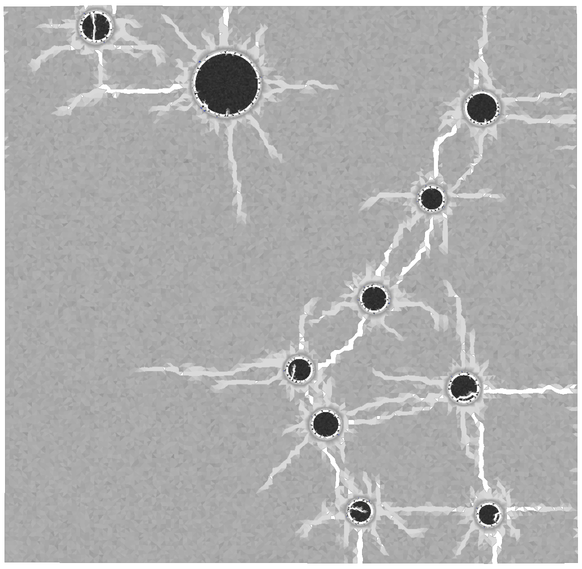


Figure 4.15: Crack patterns produced by an expansive ring of gel formed around the aggregates.

The simulations show that the macroscopic expansion and damage are strongly linked to the microstructural localisation of the reaction. Thus, the prediction of expansion from the advance of the reaction is only possible in models which simulate the direct consequences of ASR at the microstructure level.

4.6 Conclusion and perspectives

This chapter presents a physically-based model of ASR, implemented and tested in a custom-developed finite element modeling (FEM) framework. The model highlights the predominant effect of gel formation in the aggregates in the ASR degradation mechanism, notably damage in the aggregates themselves. The loss of mechanical properties can be wholly explained by the damage induced by the gel, and the model can be used to predict the evolution of mechanical properties. The model presented here showed robustness to the variation of the single fit parameter, apparent gel stiffness.

The implementation of simplified models commonly used to explain the mechanical effects of the reaction shows that they are inadequate to capture the physics of the phenomenon. The usual simplifications entail the transfer of the damage from the aggregates to the cement paste, which is experimentally observed to occur only during the advanced stages of the reaction. We also find that the shape of the free expansion/reaction curve cannot be captured by these models, unless it is imposed.

Overall, these simulations demonstrate the capacity of our finite element framework to perform well with an extremely high density of enrichments, which illustrates the robustness of our implementation.

Future work involves coupling the model with a diffusion mechanism for the alkali ions, and the introduction of time-dependent creep, which would serve as a base for a kinetic simulation of ASR.

CHAPTER 5

Implementation of Kinetics, Effect Of Confining Stress, Casting Direction and Size Effects

AMIE was successfully used to model the free expansion of mortar bars (chap. 4). In this case, the model relied on data giving the amount of reaction in the experiments corresponding to an expansion. Data from Ben Haha comparing mortars and concrete showed that this relationship depended on the particle size distribution (PSD): smaller amounts of reaction apparently caused larger expansions in concretes. However, the expansions could be renormalised to account for the aggregate content of the samples.

However, for concretes, it is impossible to acquire sufficiently large image maps to perform the analysis. Indeed, in practise, maps of 2×2 cm are the largest we could acquire in reasonable time using our back-scattered electron (BSE) microscope. Single aggregates had diameters up to 2 cm in our concrete experimental samples. Ben Haha used single images for his measures and discarded all aggregates which were not entirely contained in an image. Further, as shown in the Chap. 3.4, his protocol probably induced leaching.

Our micro-mechanical model relates expansions to gel amount. Therefore a link between the reaction and time needs to be established. The kinetics relate reaction with time based on the phenomenological observations of Ben Haha. The kinetics of gel production were calculated using data from Ben Haha on the same aggregate with the same curing conditions [1, 2]. He studied this aggregate in his thesis and recorded the reacted amount as a function of time. The reaction kinetics were only recorded for aggregates of size below 1 mm due to limitations in the image analysis software he used. However, in this study, expansion-damage measures were performed using reconstructed maps of 4 cm^2 using new software and no qualitative differences were observed between the aggregate classes[1]. As all the reactive material is identical from a mineralogical point of view, we assumed in a first approach that the kinetics were independent of the aggregate size (Fig. 5.1).

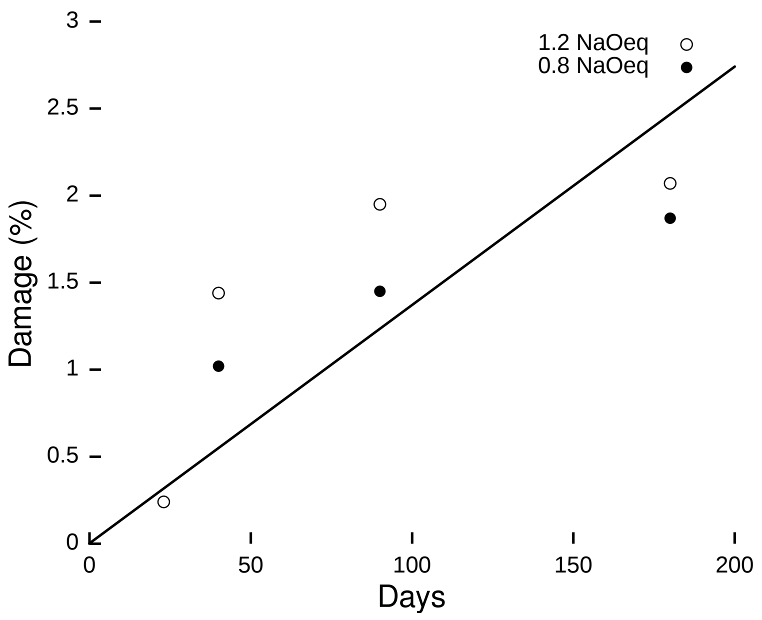


Figure 5.1: Damage with time for mortars, adapted from Ben Haha [2]. The drop in damage for the later points is probably due to leaching in Ben Haha's experimental setup. The line is the linear regression line passing through the origin.

Ben Haha found that the amount of damage was roughly linear with time (Fig. 5.1). In his protocol, samples were immersed at early age, thus facilitating early saturation. The kinetics model the gel creation, which is different from the "reaction" measured using image analysis which is a measure of damage, although we could show these were somewhat linearly related (see Chap. 4, Fig. 4.8 on p. 111). However, as stated above, due to changes in the cure, the estimation from Ben Haha's data had to be further corrected.

[1]Quantitative damage measure for large aggregates is impractical.

5.1 Effect of Sample Size, Shape and Curing Conditions

Two distinct kinds of effects were observed in our experimental data, on the kinetics and on the magnitude of the expansions. The induction period before the take-off of the expansion seem to be linked to the curing conditions and the type of samples: indeed the early part of the expansion is very sensitive to changes in sample size and preparation. However, the later expansion kinetics seem to depend on the particle size and grading, and much less on the curing conditions.

In the experimental cylinder cast with 0-16 mm concrete the sensors have a measure base of 100, and 67 mm. The 0-16 mm and the 0-8 mm concrete were with a comparable cure, but sample geometry is different. The mortar results are from Ben Haha and the cure is different. When both cure and sample size and shape are identical, it seems the induction period is also similar. This is the case for the 0-8 mm and 0-20 mm samples. If the concrete and the cure are identical such as the 0-16 mm cast cylinders and cores, the induction period is different (Fig. 5.2).

All the concrete samples were immersed after 28 days storage in a fog room. At this age, The pore network of the cement paste at 28 days is relatively closed, which will reduce the speed of diffusion of water in the samples. Thus, despite the high relative humidity, these samples were not fully saturated at the beginning of the experiment. The cored cylinders which have no wall effect, and of the mortars immersed at early age start to expand almost immediately, which may indicate that the time to initiation is due to water diffusion in the sample as the gel only expands in those parts of the samples which are saturated with water.

Table 5.1: Table of the different combinations of mix designs, sample sizes and cure tested. *Estimation from ongoing experiments.

Cure and sample type	0-4	0-8	0-16	0-20
Early immersion cast	3 days	N/A	14 days	N/A
28 days immersion cored	N/A	N/A	3 days	N/A
28 days immersion cast	150*	150	60 (cyl.)	150

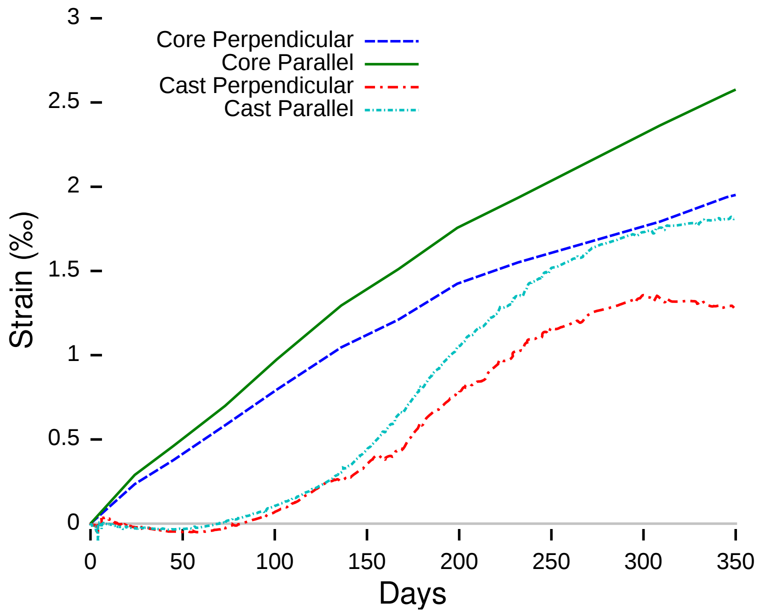


Figure 5.2: Comparison of free expansion of the cores drilled in the directions parallel and normal to the casting direction and the free expansion in the transverse and longitudinal directions of cylinders with embedded sensors. Although the mixes and curing conditions are identical, the induction period varies considerably.

Although there are considerable differences between the experiments, if the shape of the expansion curve is characteristic of the alkali-silica reaction (ASR) given the constants, aggregate and mix design, the model should capture it. If the shape cannot be captured, this means that the expansion curve is not intrinsic to the material, but depend on the sample geometry and cure.

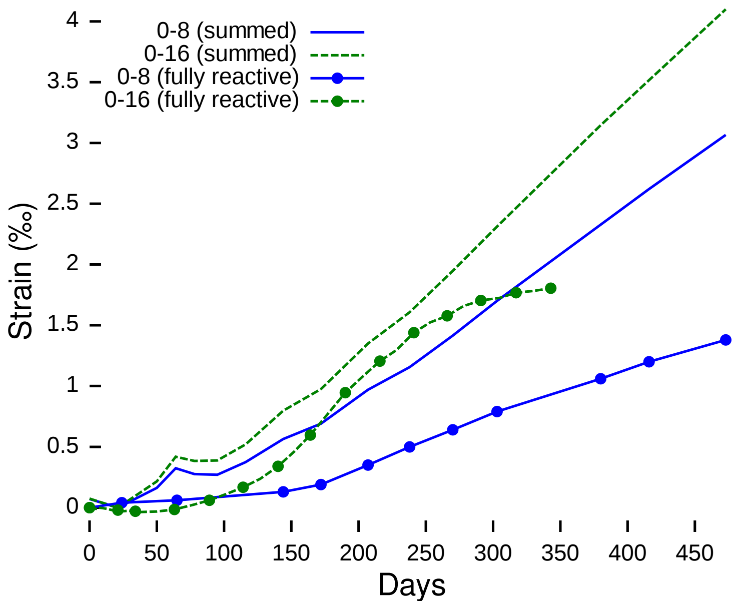
5.2 Effect Of Aggregate Size

5.2.1 Experimental Observations

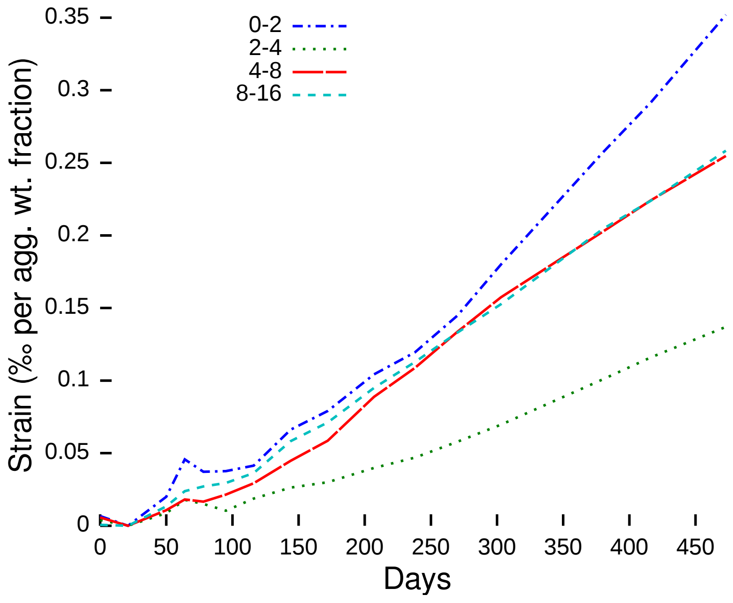
Reactive fractions

The extent of the induced expansion is not simply related to the amount of reactive material: all samples with continuous PSD have similar volumes of reactive material, ranging from 64% in mortars to around 72% for all the concretes. They however exhibit significantly different later expansions, which cannot be simply accounted for by considering the relative amount of available reactive material.

The later expansions observed in mixes with single reactive fractions are not consistent with the relative amounts of reactive materials in each mix (Fig. 5.3). Notably, the expansion potential of the 2-4 mm is much larger relative to the other fractions, which is consistent with literature reports. This indicates that the size of the individual aggregates, and thus their mode of failure is predominant when considering the effect of a particular size class. Single fractions can go on expanding without interactions and no bending of the curves is observed. This would indicate that when the concrete is fully reactive, the interactions between the different classes are not negligible.



(a) Summed expansions of fractions



(b) Expansions of fractions renormalised by reactive percentage

Figure 5.3: Comparison of the expansion of fully reactive samples and the sum of expansions of partially reactive samples. The restraining effect on the free expansion from the interactions between aggregates is apparent: after more than a year, when the expansion rates have stabilised, the expansion from the mortar part of the partly reactive samples is higher than the expansion of either of the fully reactive samples. Further, the expansions recorded are not a simple function of the mass percentage of the reactive fraction.

When the expansions of single reacting fractions are summed and compared to fully reactive samples with the same maximum diameter (D_{max}), the sum of partial expansions is systematically larger as seen in Fig. 5.3. This suggests that the interactions between reactive particles essentially inhibits the expansion. Also, the expansion of samples with single reactive fractions does not level-off like the expansion of fully reactive samples, even after 15 months. If the reaction has not been inhibited, this indicates that the interactions had an influence on the state of the microstructure.

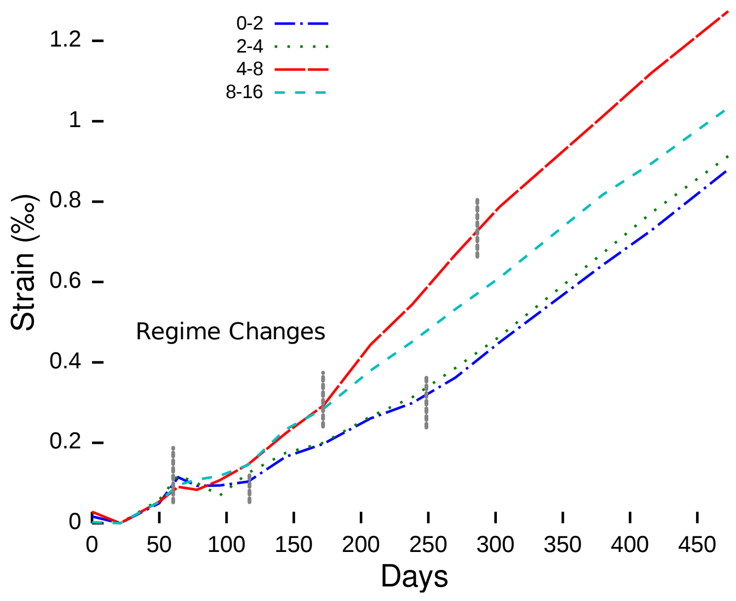


Figure 5.4: Compared expansions of samples with partly reactive aggregates.

Changes in the slope of the expansion curve observed in the reactive fraction experiment point to a size effect in the fracture behaviour of the aggregates (Fig. 5.4). Larger expansions can occur when confinement is lowered, such as when aggregates or paste fail by cracking. Because the mineralogy is kept constant across all aggregate sizes, the differences in modes of cracking can only come from size effects. A smaller aggregate is more brittle than a larger one from the point of view of a gel pocket. Therefore, smaller aggregates should crack earlier than larger ones. However, further propagation of cracks in the paste is driven by a combination of the further expansion of the aggregate, the disposition of the gel pockets and the state of damage of the aggregate. In general, it can be be assumed that propagation will be larger from aggregates which have an intermediate size: a large aggregate will contain the gel expansion and a small aggregate will be too damaged to exert much pressure.

5.2.2 Simulations

AMIE was used to simulate the effect of individual fractions in a concrete. The numerical setup considered fully reproduced microstructures. A 2D microstructure from a slice of a 3D microstructure was not used because this would over represent larger aggregate fractions [112, 113]. To reproduce the interactions between the different aggregate classes, we have decided to transform each aggregate in a generated PSD to its expected diameter in a slice. This way, the PSD in the 2D setup is equivalent to the one in the 3D in terms of neighbourhoods of differently sized aggregates. Furthermore, the same amount of expansion comes from the same aggregate class, as the fractions are proportionately kept, and the effects of size and in-aggregate crack propagation are reproduced (Fig. 5.5).

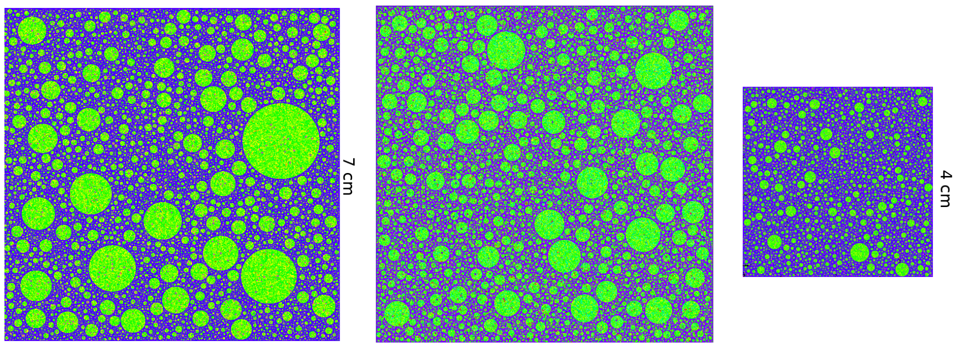


Figure 5.5: Simulated slices for 0-8, 0-16 and 0-4 aggregate sizes.

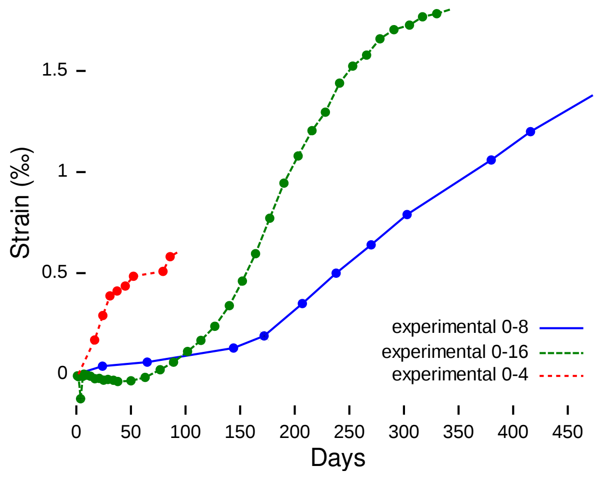
In the simulations, the sample size was 40 × 40 mm for the mortar and 70 × 70 mm for the concretes. In both cases, this size is sufficient for the numerical samples to be representative volumes. Since these sizes are also identical to the experimental mortars and 0-8 mm concrete cases, they will, in these cases, reproduce the mechanical effects of the wall effect where the material at the edge of the specimen contains no aggregates.

Effect of the D_{max}

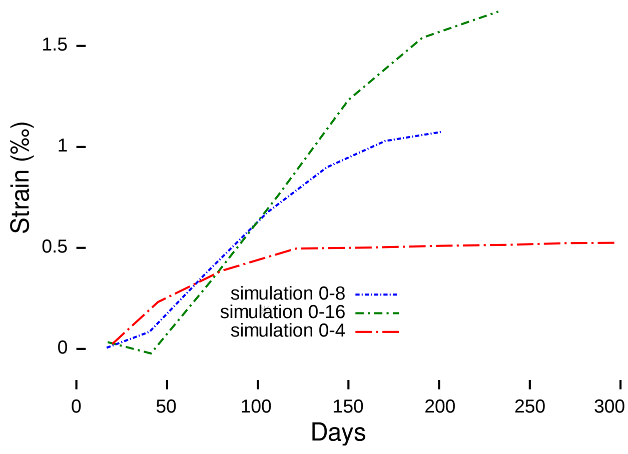
To test the kinetic model, simulations were run with the same PSD as the experiments. The effect of fully reactive PSD with varying D_{max} was tested with 0-4, 0-8 and 0-16 mm numerical samples.

The kinetic parameter used for the following simulations was that 1% of aggregate volume is transformed into gel in 170 days at 40 °C and constant $0.135\,\mathrm{mol\,l^{-1}}$ Of NaOH in curing solution. This kinetic parameter is fixed from Ben Haha's measures, but corrected for the damage/reaction correlation which was obtained from simulations in Chap. 4 and for the leaching effect in Ben Haha's experiments, as discussed above.

The comparison of time-expansion curves, experimental and simulated is seen on Fig. 5.6 which shows the curves predicted by the model if the expansions start immediately.



(a) Experiments



(b) Simulations

Figure 5.6: Compared expansions and reaction of numerical and experimental samples of different D_{max}. The order in which the expansions take-off is conserved by the simulations. The differences in time for the take-off can be explained by differences in protocol and sample size.

The starting time of the expansions in the simulation was then fitted to coincide with that of the experiments. The gel production kinetics are unchanged. As an offset was applied to match the initiation of the expansion, the order in which the curves start their expansion was affected (Fig. 5.7).

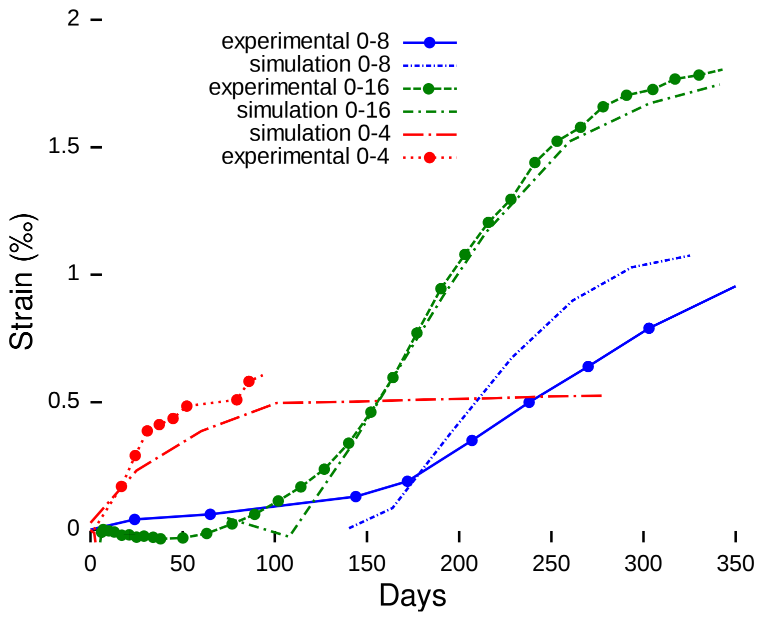


Figure 5.7: Effect of D_{max}. The order and magnitude of the measured extensions agree well between the experiments and the simulations. The kinetic factor for the rate of gel production is the same in all experiments. The simulated times have been offset to match the initiation of the expansion.

As can be seen in Fig. 5.7, the simulations capture the magnitude of the expansion, as well as the general shape of the curve. However, the kinetics are not captured perfectly, and the simulations diverge from the experiments. Notably, the 0-8 mm concrete which has the longest induction time has the worst fit. It was then assumed that samples of different shapes and mix designs did not react at the same rate.

To account for the differences between the samples, a third kinetic fit was developed where two kinetic factors are fitted for each experimental condition, the delay and the rate of reaction. The values obtained for the delay are reported in the table below and the factors are plotted in Fig. 5.8.

Table 5.2: Kinetic factors for the simulations.

Factor	0-4	0-8	016
Delay	-10	90	70
Acceleration	100	300	170

The inverse of the acceleration factor, in days per percent, is perfectly linearly correlated to the delay. This may indicate that the two fit factors are linked.

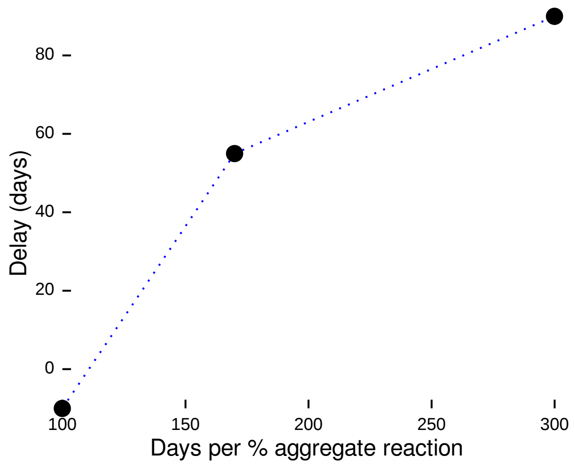


Figure 5.8: Kinetic factors for the best fit of the simulated time-expansion curves for various PSD. Each PSD had two parameters adjusted, the delay before the initial expansion and the time needed to transform a fraction of the aggregate into gel (labelled "factor").

If the kinetic factors are varied for each condition, the fits can be considerably improved as shown on Fig. 5.9. Indeed, a near perfect fit can be obtained which captures the shape of the expansion as well as its magnitude.

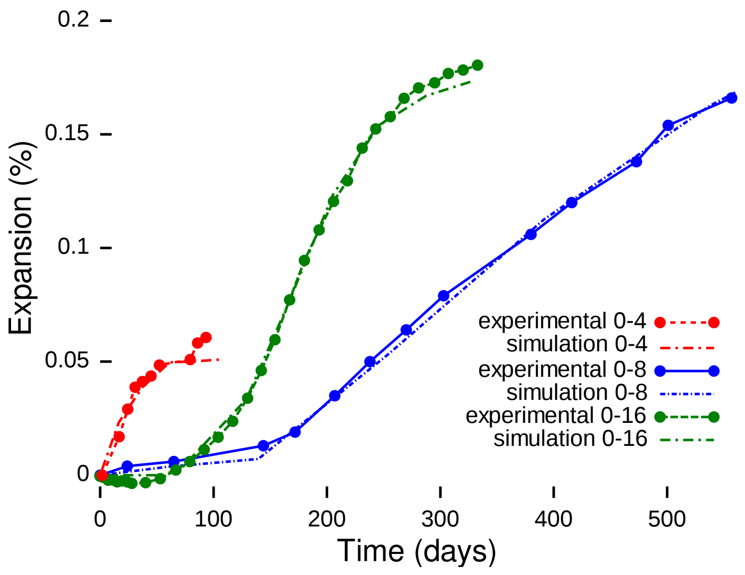


Figure 5.9: Effect of D_{max}. The order and magnitude of the measured extensions agree well between the experiments and the simulations. The kinetic factor for the rate of gel production is different in all experiments. The simulated times have been offset to match the initiation of the expansion.

The 0-8 mm simulation and experiments diverge when approximately 1 % of the aggregate mass has reacted. The simulated expansions level off earlier than in the experiments. This is probably because the limit of 3% reactive material per aggregate, discussed in Chap. 4 and above is too low. The 0-16 mm simulation does not display this behaviour because the setup has been modified to allow greater amounts of reactions in individual aggregates.

These simulations validated the simple kinetics implemented. As the shape of the expansion curve could be reproduced by the simulation given a simple linear kinetic model, this indicates that the shape of the curve is indeed characteristic of the PSD and aggregate.

Individual Fractions

Further simulations were run which reproduced the experimental conditions where either only the 2-4 mm or the 4-8 mm class is reactive. The results are reported on Fig. 5.10.

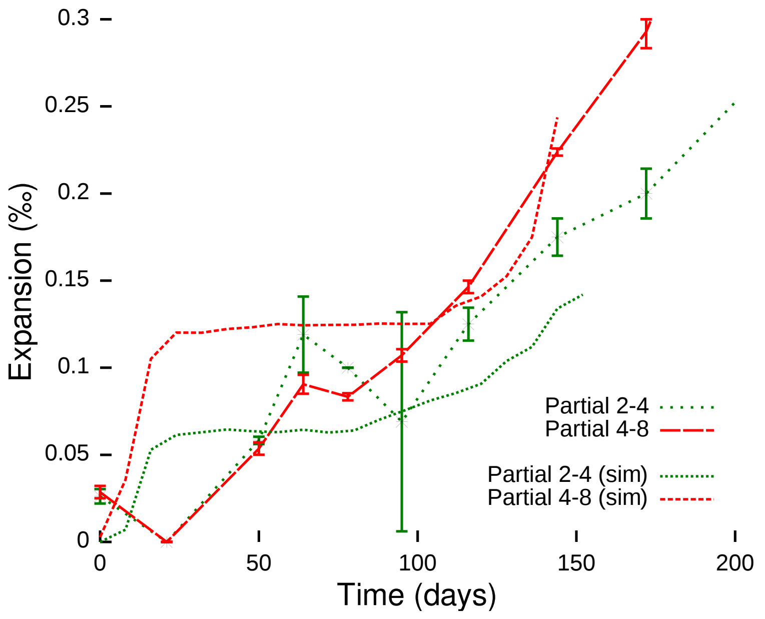


Figure 5.10: Experimental and simulated expansions of samples with single reactive fractions.

The simulations agree reasonably well with the experimental results. In the simulations a plateau is apparent. This plateau is caused by the aggregates having already cracked but the amount of reaction is for a period insufficient to cause propagation of cracks in the paste. There is some sign of this phenomenon in the experiments but further studies are needed to confirm it.

The plateau observed is consistent with the phenomenological model proposed for the expansion-curve: elastic expansion, aggregate cracking, gel build-up in the aggregates, paste cracking. The gel build-up in the aggregates manifests itself by a pause in the expansion. However, the duration of the gel build-up depends on the interactions with other reactive aggregates.

Although the fully reactive sample is much more damaged than the partly reactive one, it has stopped expanding. On the contrary, the partially reactive sample has entered a linear regime of expansion. Observation of the damage state of fully reactive and partially reactive samples at the end of the simulations (Fig. 5.11) provides a possible explanation for the differences of shapes of expansion curves for partly reactive and fully reactive samples.

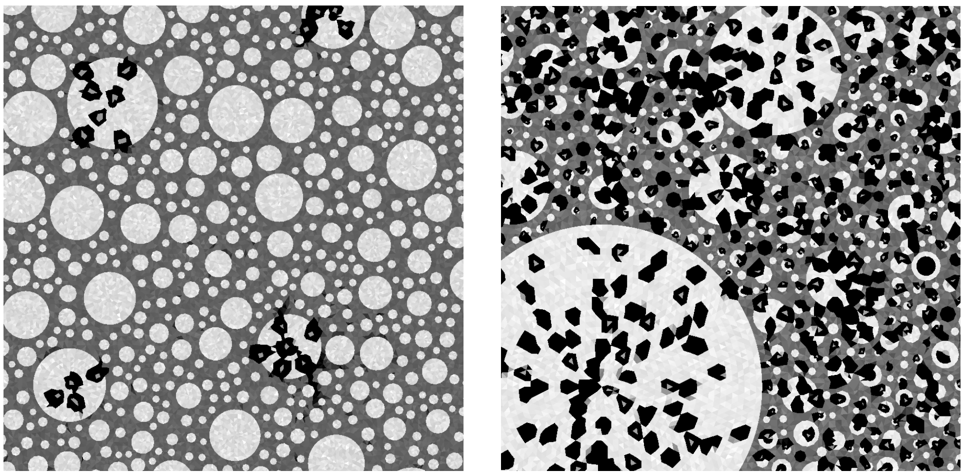

(a) Partially reactive (b) fully reactive

Figure 5.11: Damage state at the end of simulations for partially reactive and fully reactive concretes. Although the fully reactive sample is much more damaged than the partly reactive one, it has stopped expanding. On the contrary, the partially reactive sample has entered a linear regime of expansion. This slowdown can be explained because for expansion to occur, the samples must retain enough stiffness, and the fully reactive sample has reached a damage state where further expansion of the gel is not constrained anymore and therefore does not translate to further macroscopic expansion.

The early expansion behaviour seems to be determined mostly by the individual aggregate sizes. This could be explained by the fact that all aggregates act independently, and the gel is itself restrained in the aggregates. Once the aggregates have started cracking, the apparent stiffness of the aggregates contrasts less with that of the paste. Thus the interactions between the aggregates become important. This could explain the experimentally observed curving down of the expansion, frequently interpreted as the exhaustion of the reactive material. This can be explained because for expansion to occur, the samples must retain enough stiffness, and the fully reactive sample has reached a damage state where further expansion of the gel is not constrained anymore and therefore does not translate to further macroscopic expansion.

5.3 Confining Stress

We used AMIE to compute expansions under loaded conditions using a setup similar to the one used for the free deformation simulations (Fig. 5.12a). In a first set of simulations, a single highly detailed aggregate was considered under free and loaded conditions. These simple simulations were used to study the effect of the load in crack propagation in the matrix and the aggregates, as well as the interactions between the gel pockets within an aggregate. In a second set of simulations, samples with full PSD were considered under various loading conditions around the experimentally observed transition in terms of expansion regimes.

5.3.1 Single Inclusions

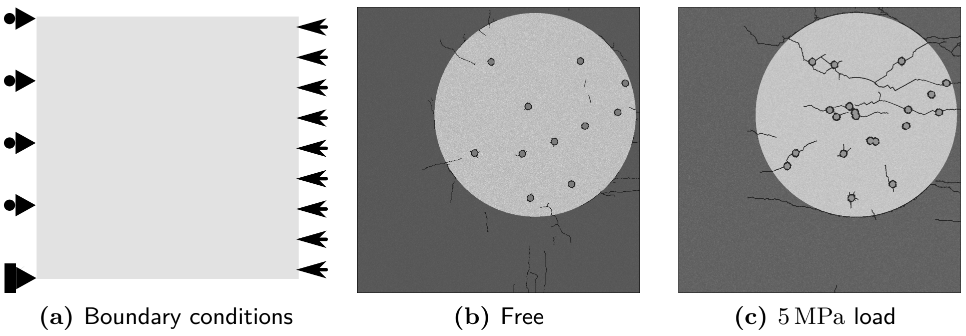

(a) Boundary conditions (b) Free (c) 5 MPa load

Figure 5.12: Crack pattern of a numerical sample with a single inclusion in unloaded and loaded conditions. The cracks are mainly in the paste (dark grey), which is in tension as there are no other expanding aggregates to keep it in compression. The aggregate (light grey) is partly cracked, around the gel pockets, but also from cracks which propagated from the paste.

With a single inclusion, expansion puts the paste around it into in tension. This causes cracks to initiate preferentially in the paste. In the unloaded condition, no particular direction of propagation is favoured by the cracks, except for the normal tendency to propagate in directions normal to interfaces. The cracks seem to initiate more frequently in the paste and then propagate into the aggregate. Under the 5 MPa load, the cracking is more pronounced and the cracks initiate more frequently in the aggregate. Under load, a clear direction of propagation is favoured: parallel to the load.

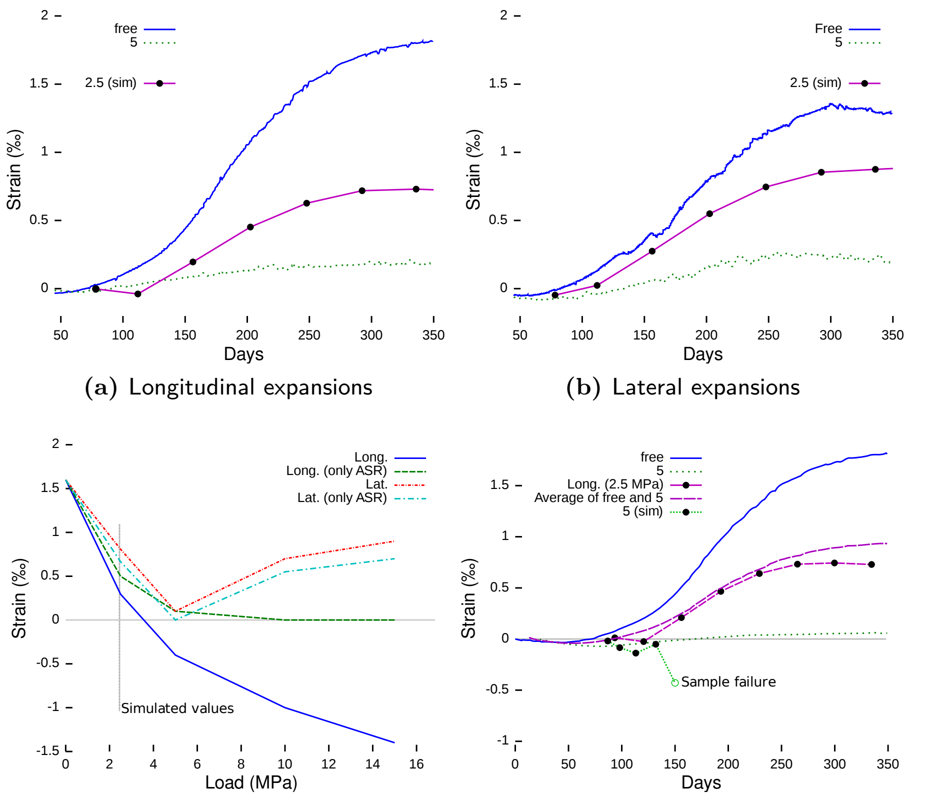
This experiment showed that even at the lowest load applied in the experimental setup, a considerable effect could be expected in terms of the cracking behaviour of the material at the microstructure level. The externally applied load imposes the direction of most cracks. The most striking consequence is that aggregates are more prone to splitting along the load axis (Fig. 5.12). The stress around the gel pockets, calculated by numerical simulation is in the order of 10 GPa, and the imposed stress on the samples is therefore negligible with respect to the criterion for crack propagation and initiation. However, the stress and strain fields are strongly oriented by the load and will direct the direction of propagation.

5.3.2 Full PSD

The second set of experiments, with full PSD, considered loads around the critical 5 MPa value. 5 MPa was considered a critical value because our experiments showed that under this load, the expansion due to ASR is scaled down from the free expansion and not transformed in terms of shape, indicating that the transition between damage regimes is not complete. Simulations of expansions under loads of 2.5 MPa, 5 MPa and 7.5 MPa were run.

The simulated expansions at a load of 2.5 MPa (Fig. 5.13) are well in the trend of the observed effect of stress for loads under 5 MPa. This indicates that for such loads, although the direction of cracking is affected, as demonstrated by lower longitudinal and lateral expansions than the average between the experimental 5 MPa and free expansion curves, the mode of damage remains the same: the shape of the expansion curve is not affected. When the final expansion reported by this simulation is plotted along with the experimental values, it falls very clearly in the trend (Fig. 5.13c).

The crack patterns for the loads above 2.5 MPa were significantly denser than those observed for simulations of free expansions (Fig. 5.14). The cracks also tend to be oriented in the direction of the load. This indicated that the load could indeed enhance the damaging consequences of the ASR.



(a) Longitudinal expansions

(b) Lateral expansions

(c) Final expansions recorded as a function of the load.

(d) Comparison of $2.5\,\mathrm{MPa}$ and $5\,\mathrm{MPa}$

Figure 5.13: Simulated expansion under a $2.5\,\mathrm{MPa}$ load. As expected, the lateral expansion (a) is slightly lower than the average between the $5\,\mathrm{MPa}$ and the free case, as for the lateral (b). (c) Final expansions recorded as a function of the load. The point at $2.5\,\mathrm{MPa}$ is simulated and fits perfectly in the different trends, thus validating the model for loads under $5\,\mathrm{MPa}$. (d) Comparison of the expansion under $2.5\,\mathrm{MPa}$ and $5\,\mathrm{MPa}$ loads and with experiments.

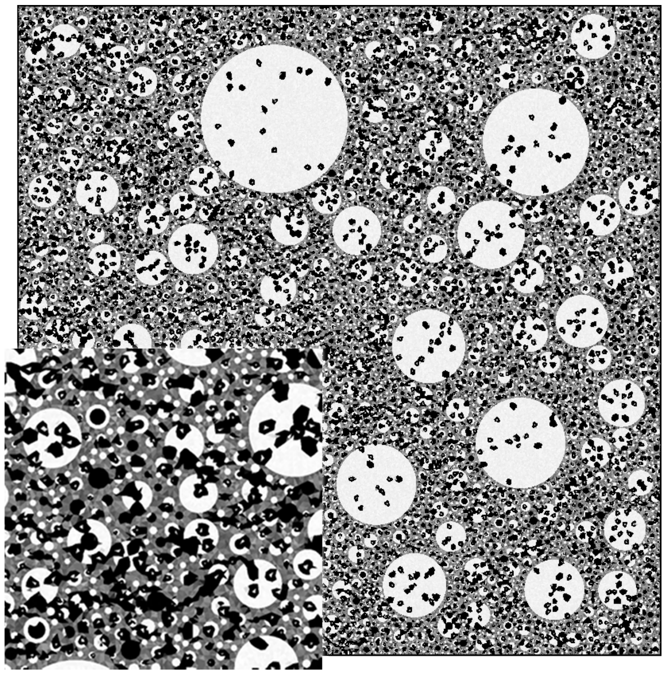


Figure 5.14: State of the sample under 2.5 MPa load at the end of the simulation. The cracks are, as in the single inclusion simulations, on average oriented in the direction of the load ($\rightarrow\leftarrow$)

At higher loads, the damage-enhancing effect is more pronounced as visible on Fig. 5.15 and leads to catastrophic failure of the simulated samples. With a simulated gel stiffness value of 0.7 times that of calcium silicate hydrate (C-S-H), the simulated samples undergo catastrophic failure after 125 simulated days for a 5 MPa load and after 90 simulated days for the 7.5 MPa load. These times were obtained by assuming 1% of the aggregates are transformed into gel in 210 days, which is the same kinetic parameter used for the free expansions of 0-16 mm samples. Fig. 5.15 illustrates the state of the samples when they fail. The simulations diverge at the same time as the experiments undergo an acceleration of their lateral expansions. At the time of failure, the simulated samples had essentially split and as a slip condition had been set on the boundary, they became subjected to solid body motion[2]. Indeed, the damage model used does not account cracks closing in compression, and this may explain the failure of the model to capture the behaviour at later stages of the reaction.

The experimentally observed acceleration of damage may correspond to its accumulation. The simulations perhaps capture the change of mode in

[2]This explains the absence of convergence as no unique solution for displacement exists.

the damage evolution, but the damage model used is inadequate to capture the sample behaviour under these conditions.

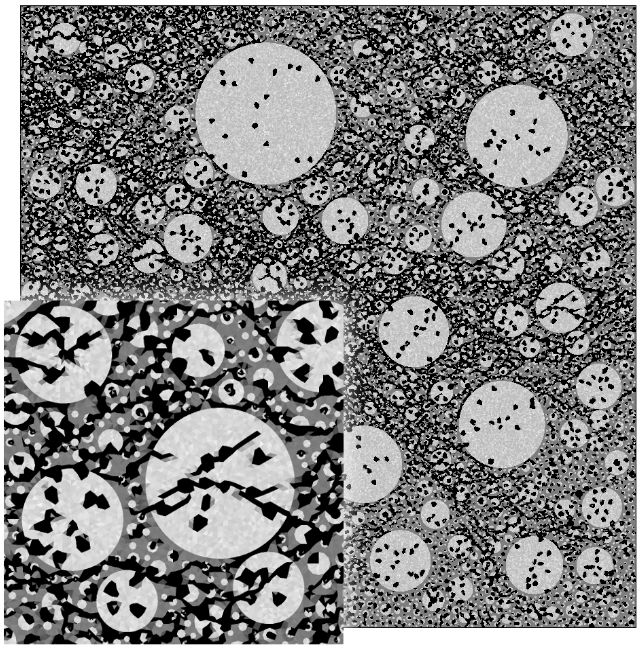


Figure 5.15: State of the sample under 5 MPa load at failure. The sample has split and the simulation diverged. Direction of load: $\rightarrow\leftarrow$

Further experiments were performed with a gel stiffness value of 0.6 and 0.5 times that of C-S-H to verify whether the cracking was induced by excessively high stresses induced by the gel pockets, but this did not affect significantly the results. The free expansion modelling of mortars (Chap. 4) had shown that the final expansion was not very sensitive to variation in the stiffness of the gel, and this was confirmed. Even with the softer gel, the samples underwent catastrophic failure early in the simulation (Fig. 5.13d).

At larger loads, experimentally, the samples do not exhibit any longitudinal expansion at all. This means that all the expansion due to the gel in that direction is contained. This in turn hints that the cracks must almost all be in planes parallel to the direction of the loading. In two dimensional simulations, this will always lead to the sample splitting, as soon as a single line of damaged elements joins the two sides of the sample, which will occur early in the simulation. To prevent this, displacements normal to the axis of loading could be blocked on the loaded boundaries of the sample. However, doing that would not reproduce the experimental boundary conditions.

The model does not account for gel flow, which might be the cause of the experimentally observed behaviour. Notably, gel pressure may be relieved by flow more efficiently than by crack opening only.

5.4 Conclusion and Perspectives

The ASR micro-mechanical model was validated for a range of aggregate sizes and in the case of single reactive fractions. The effects of size were related first to the failure behaviour of individual aggregates, and further to the global expansion potential of larger aggregates. Experiments with single reactive aggregate size classes highlighted the importance of the interactions between the individual aggregates. These experimental results were also used to validate a kinetic model for the development of ASR.

The kinetics of the expansion seem to be affected by the sample size. Although a simple linear model could be used to reproduce the expansion curves, indicating that these are intrinsic to the material, further experiments are required to better understand the link between sample size and geometry.

Experimental work on the effect of restraining stress showed that the load causes a change in the damage evolution of ASR affected concrete, probably enhancing the damaging effect of the reaction. This was confirmed by modelling: the uniaxial load imposes a propagation direction for the cracks as the reaction develops. Simulations of 2.5 MPa load seemed to give reasonable results but the present form of the model led to samples failing catastrophically at loads of 5 MPa and higher. The variation of the time of the failure was found to correlate well with the experimentally observed acceleration of strain development under loads higher than 5 MPa. Taken together, these results show the need for the development of a more elaborate fracture behaviour model which better accounts for crack behaviour under compression.

Conclusion

This thesis followed the work of Ben Haha who formulated a hypothesis linking the damage state of the reactive aggregates and the macroscopic expansion of samples. This hypothesis was tested on a range of different aggregates using a specially developed software and could be verified. Although testing with a larger set of aggregates showed the expansion-damage curve to exhibit more variation, the general trend was very well conserved. This served as the basis for the formulation of a numerical micro-mechanical model.

To simulate the ASR at the microstructure level a generic finite element modeling (FEM) framework was developed and a novel method for computing damage was implemented. This new framework made it possible to formulate a model with highly detailed representation of the individual aggregates and gel pockets. This allowed the model to take into account the interactions between individual aggregates as well as the effect of transformation into expansive gel in individual aggregates. It was shown that it was possible to model ASR using finite elements using very few fitting parameters. We found that to model the mechanical consequences of the reaction which result from a given percentage of reaction, a single fit parameter was required, which accounts for the mechanical properties of the ASR gel, which are known to depend on the availability of Ca^{2+} and K^{+} in the pore solution, and therefore on the aggregate type/cement type.

A kinetic model was developed which allowed the fitting of the time-expansion behaviour of the experimental samples with different mix design parameters, shape and sizes. Only two parameters, the amount of gel produced per time and the delay to the onset of the reaction need to be fitted as they depend on the geometry of the member considered and the curing conditions. At present, these two parameters are empirical, further research is required to understand their causes. In our simulations, they however seemed related, indicating that they might be two aspects of the same phenomenon, probably linked to water diffusion in the sample.

The effects of mix design parameters and loading conditions were experimentally tested in this thesis. It was found that the effects of the PSD on the shape of the S-shaped curve characteristic of this reaction and the final expansion are caused by the interactions between the expanding aggregates. Larger D_{max} results in larger final expansion, but also slower take-off. The effect of restraining stress is a reduction of the total expansion if it is below 5 MPa. For stresses beyond that, the expansions are still lower than the free expansions, but occur faster as the load increases and become larger with the load. This indicated that the load enhanced the damaging effect of ASR. It was possible to model the effect of stress for the lower loads. However, the simulations did not converge in the case of the larger loads, indicating that a 3D model of the damage is probably required.

Future developments involve running simulations in 3D to better understand to role of multiaxial loading in the degradation of the material. The integration of a diffusion model is also necessary to verify the hypothesis that the delay of the reaction as well as the kinetics are indeed linked to the diffusion of water in the samples.

Acknowledgements

The Swiss Federal Office for Energy (OFEN) is gratefully acknowledged for their support, financial and otherwise.

I wish to thank my thesis supervisor, Karen, for trusting me and letting build this simulation framework and move the project in the direction I wished, despite the not obvious benefits. Thanks for all the discussions, scientific and otherwise which were had, around cups of coffees, glasses of wine and mugs of beer. They are what made the “cup of tea factor” one of the more pleasant aspect of this thesis.

I wish to thank Amor Guidoum and Philippe Vulliemin for their help setting up and running the experiments in this thesis, it would not have been possible without them. Un grand merci à M. Diserens, notre mécanicien-magicien, pour son aide au début de ma thèse. Merci à Lionel de faire tourner les machines en bas, et d’être toujours là pour résoudre nos problèmes pratiques.

I will keep fond memories, and I hope lasting friendship with my LMC colleagues. A lab can be more than a place where one works, it can also be a second home. I have particular respect for those of you who, over all these years made a particular effort to keep the social cohesion of the lab, be it through the organisation of events and outings, or more prosaically, the lab-cleaning days and the day-to-day fixing and maintenance of the apparatuses. So, Momo, Juju, Shanky, Manu[3], Vaness’, Patrick, Christophe, Théo, Alex, Adityia, Aude, Lionel[4], and all the new ones I have not yet

[3]Gallucci.

[4]Order as a function of office location and time of arrival in the lab.

gotten to know so much, thank you. And keep the LMC such a wonderful place. And keep the barbecue tradition alive.

Merci à Stéphane et Manu[5] pour les déjeuners[6] durant toutes ces années à l'EPFL. Merci à Cécile pour les sorties/soirées qu'elle a organisées et animées.

Merci à Elisa pour son amour et son aide dans la rédaction de cette thèse. Pour toutes ces années ensemble, et pour partager ma vie. J'espère que tu continueras à la partager toujours.

[5]Eckard.

[6]≪ Dîners ≫ pour Steph.

Part III

Bibliography and Supplementary Material

Bibliography

[1] Mohsen ben Haha, Emmanuel Gallucci, Amor Guidoum, and Karen L. Scrivener. Relation of expansion due to alkali silica reaction to the degree of reaction measured by sem image analysis. *Cement and Concrete Research*, 37(8):1206–1214, August 2007.

[2] Mohsen ben Haha. *Mechanical Effects of Alkali Silica Reaction in concrete studied by SEM-image analysis.* PhD thesis, École Polytechnique Fédérale de Lausanne, May 2006.

[3] T.E. Stanton. Studies to develop an accelerated test procedure for the detection of adversely reactive cement-aggregate combinations. In *ASTM Proceedings*, pages 875–893, 1943.

[4] P. Léger, P. Côté, and R. Tinawi. Finite element analysis of concrete swelling due to alkali-aggregate reactions in dams. *Computers & Structures*, 60(4):601 – 611, 1996.

[5] Victor Saouma and Luidgi Perotti. Alkali-aggregate reactions in dams; stress analysis and long term predictions. In *ASDSO Dam Safety Conference*, pages 601 – 611, September 2005.

[6] Stéphane Multon, N. Leklou, and L. Petit. Coupled effects of aggregate size and alkali content on asr expansion. *Cement and Concrete Research*, 38(3):350–359, 2008.

[7] J. M. Ponce and O. R. Batic. Different manifestations of the alkali-silica reaction in concrete according to the reaction kinetics of the reactive aggregate. *Cement and Concrete Research*, 36(6):1148–1156, 2006.

[8] Stéphane Multon and François Toutlemonde. Effect of applied stresses on alkali–silica reaction-induced expansions. *Cement and Concrete Research*, 36(5):912–920, 2006.

[9] Adil Binal. The determination of gel swelling pressure of reactive aggregates by asgpm device and a new reactive-innocuous aggregate decision chart. *Constr Build Mater*, 22(1):1 – 13, January 2008.

[10] Mitsunori Kawamura and Kazuma Iwahori. Asr gel composition and expansive pressure in mortars under restraint. *Cement & Concrete Composites*, 26(1):47 – 56, 2004.

[11] C. E. Tambelli, J. F. Schneider, N. P. Hasparyk, and Paulo J. M. Monteiro. Study of the structure of alkali–silica reaction gel by high-resolution NMR spectroscopy. *Journal of Non-Crystalline Solids*, 352(32):3429–3436, August 2006.

[12] Catherine Larive. *Moyens et Méthodes d'Essai*, chapter 3. Études et recherches des laboratoires des ponts et chaussées, 1998.

[13] Niza Smaoui, Marc-André Bérubé, Benoît Fournier, and Benoît Bissonnette. Influence of Specimen Geometry, Orientation of Casting Plane, and Mode of Concrete Consolidation on Expansion Due to ASR. *Cement, Concrete, and Aggregates*, 26(2):1–13, 2004.

[14] I. Mohamed, L. Curtil, S. Ronel-ldrissi, and P. Hamelin. Influence of composite materials confinement on alkali-aggregate expansion. *Materials and Structures*, 36:387–394, 2005.

[15] Stéphane Poyet. *Étude de la dégradation des ouvrages en béton atteints par la réaction alcali-silice : Approche expérimentale et modélisation numérique multi-échelles des dégradations dans un environnement hydro-chemo-mécanique variable*. PhD thesis, Université de Marne la Vallée, 2003.

[16] B. G. Galerkin. Series solution of some problems of elastic equilibrium of rods and plates. *Vestn. Inzh. Tech*, 19:897–908, 1915.

[17] R. Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bull. Amer. Math. Soc.*, 49(1):1–23, 1943.

[18] A. Hrennikoff. Solution of problems in elasticity by the framework method. *Journal of Applied Mechanics*, 8(4):169–175, 1941.

[19] Guido van Rossum. python. On The Web `http://www.python.org/`, 2000.

[20] Philippe Montarnal, Alain Dimier, Estelle Deville, Erwan Adam, Jérôme Gaombalet, Alain Bengaouer, Laurent Loth, and Clément Chavant. Coupling methodology within the software platform alliances. In M. Papadrakakis, E. Oñate, and B. Schrefler, editors, *Coupled Problems 2005*, 2005.

[21] Hibbitt, Karlsson, Sorenson (HKS), and Pawtucket. Abaqus, 2003. http://www.hks.com.

[22] G. Rio, H. laurent, and G. Blès. Asynchronous interface between a finite element commercial software abaqus and an academic research code herezh++. *Advances in Engineering Software*, 39(12):1010–1022, 2008.

[23] C. Timbrell, P. Claydon, and G. Cook. Application of abaqus to analysis of 3d cracks and fatigue crack growth prediction. In *Proccedings of the 1994 ABAQUS Users' Conference, Rhode Island, U.S.A.* Zentech International Limited, 103 Mytchett Road, Camberley, Surrey, GU16 6ES, U.K., 1994.

[24] J. Sladek, V. Sladek, and S. N. Atluri. Local boundary integral equation (lbie) method for solving problems of elasticity with nonhomogeneous material properties. *Computational Mechanics*, 24(6):456–462, January 2000. .

[25] T. Strouboulis, K. Copps, and I. Babuška. The generalized finite element method : An example of its implementation and illustration of its performance. *International Journ Numer Meth. Engng*, 47(8):1401–1417, 2000.

[26] Vinh Phu Nguyen, Timon Rabczuk, Stéphane Bordas, and Marc Duflot. Meshless methods: a review and computer implementation aspects. *Mathematics and computers in Simulation*, 79:763–813, 2008.

[27] Jean-François Remacle, B. K. Karamete, and M. Shephard. Algorithm oriented mesh database. In *Ninth International Meshing Roundtable, Newport Beach, California*, pages 349—359, October 2000.

[28] C. Cobb, W.D. Rolph, and D. Wilmarth. System architectures for support of nonlinear finite element analysis. *Computers and Structures*, 40(2):451–461, 1991.

[29] Billy Fredriksson, Jaroslav Mackerle, and B. G. Allan Persson. Finite-element programs in integrated software for structural mechanics and CAD. *Computer-aided Design*, 13(1):27–39, 1981.

[30] Hung-Ming Chen and Yu-Chin Lin. Web-fem: An internet-based finite-element analysis framework with 3d graphics and parallel computing environment. *Advances in Engineering Software*, 39(1):55–68, 2008.

[31] Jun Peng, David Liu, and Kincho H. Law. An engineering data access system for a finite element program. *Advances in Engineering Software*, 34(3):163–181, 2003.

[32] R. I. Mackie. An object-oriented approach to fully interactive finite element software. *Advances in Engineering Software*, 29(2):41–46, 1998.

[33] T. J. R. Hughes, J. A. Cottrell, and Y Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Comput. Methods Appl. Mech. Engrg.*, 194:4135–4195, 2005.

[34] M. D. Rucki and G.R. Miller. An adaptable finite element modelling kernel. *Computers and Structures*, 69:399–409, 1998.

[35] Lichao Yu and Ashok V. Kumar. An object-oriented modular framework for implementing the modular framework finite element method. *Computers and Structures*, 79:919–928, 2001.

[36] Yves Dubois-Pèlerin and Thomas Zimmermann. Object-oriented finite element programming: Iii. an efficient implementation in c++. *Comput. Methods Appl. Mech. Engrg.*, 108(1):165–183, 1993.

[37] Thomas Zimmermann, Yves Dubois-Pèlerin, and P. Bomme. Object oriented finite element programming: I. governing principles. *Comput. Methods Appl. Mech. Engrg.*, 93(8):291–303, 1992.

[38] Stéphane Bordas, Vinh Phu Nguyen, Cyrille Dunant, Hung Nguyen-Dang, and Amor Guidoum. An object-oriented extended finite element library. *International Journ Numer Meth. Engng*, 71(6):703–732, 2007.

[39] J. Besson and R. Foerch. Large scale object-oriented finite element code design. *Comput. Methods Appl. Mech. Engrg.*, 142(1):165–187, 1997.

[40] Ted Belytschko, Y. Y. Lu, and L. Gu. Element-free Galerkin methods. *International Journ Numer Meth. Engng*, 37(2):229–256, 1994.

[41] Cyrille Dunant, Phu Nguyen Vinh, Mourad Belgasmia, Stéphane Bordas, and Amor Guidoum. Architecture tradeoffs of integrating a mesh generator to partition of unity enriched object-oriented finite element software. *Revue Européenne de Mécanique Numérique*, 16:237–258, March 2007.

[42] J. Mackerle. Object-oriented techniques in fem and bem, a bibliography (1996-1999). *Finite Elements in Analysis and Design*, 36:189—196, 2000.

[43] J. A. Sethian. *Level Set Methods & Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science.* Cambridge University Press, Cambridge, UK, 1999.

[44] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409–436, 1952.

[45] L. Grippo and S. Lucidi. A globally convergent version of the Polak-Ribiere conjugate gradient method. *Mathematical Programming*, 78(3):375–391, 1997.

[46] H. van der Vorst. BI-CGSTAB: a fast and smoothly converging variant of BI-CG for the solution of non-symmetric linear systems. *SIAM J. Sci. Stat. Comput*, 13:631–644, 1992.

[47] Nicolas Moës, M. Cloirec, P. Cartraud, and Jean-François Remacle. A computational approach to handle complex microstructure geometries. *Comput. Methods Appl. Mech. Engrg.*, 192(29):3163–3177, 2003.

[48] Various. Pov ray. On The Web : `http://www.povray.org/`.

[49] S. Magnenat and L.-O. Charrière. Globulation 2. On The Web `http://glob2.ysagoon.com`, 2000.

[50] S. Magnenat. Enki. On The Web `http://teem.epfl.ch`, 2000.

[51] Greg N. Frederickson. Optimal algorithms for tree partitioning. In *Proceedings of the Symposium on Discrete Algorithms*, pages 169–177, 1991.

[52] O. Devillers, S. Meiser, and M. Teillaud. Fully dynamic delaunay triangulation in logarithmic expected time per operation. *Comput. Geom. Theory Appl.*, 2(2):55–80, 1992.

[53] Christophe Geuzaine and Jean-François Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 2008.

[54] P. Laborde, J. Pommier, Y. Renard, and M. Salaun. High order extended finite element method for cracked domains. *International Journ Numer Meth. Engng*, 190(47):6183–6200, 2004.

[55] Stéphane Bordas and Marc Duflot. A-posteriori error estimation for enriched finite element methods. *Computer Methods in Applied Mechanics and Engineering*, 196(35):3381–3399, 2007.

[56] Giang D. Nguyen. A thermodynamic approach to non-local damage modelling of concrete. *International Journal of Solids and Structures*, 45(7):1918–1934, November 2007.

[57] Isabelle Peyrot, Pierre Olivier Bouchard, François Bay, Fabrice Bernard, and Eric Garcia-Diaz. Numerical aspects of a problem with damage to simulate mechanical behavior of a quasi-brittle material. *Computational Materials Science*, 40(3):327–340, April 2007.

[58] Predeleanu. Modeling dynamic strain localization in inelastic solids. *Journal of Materials Processing Technology*, 32(1):75–89, 1992.

[59] J. Bessona, D. Steglichb, and W. Brocks. Modeling of plane strain ductile rupture. *International Journal of Plasticity*, 19(10):1517–1541, April 2003.

[60] G. A. Francfort and J.-J. Marigo. Revisiting brittle fracture as an energy minimisation problem. *J. Mech. Phys. Solids*, 46(8):1319–1342, January 1998.

[61] Dusan Krajcinovic. Damage mechanics. *Mechanics of Materials*, 8(2):117–197, March 1989.

[62] Matthew J. DeJong, Max A.N. Hendriks, and Jan G. Rots. Sequentially linear analysis of fracture under non-proportional loading. *Engineering Fracture Mechanics*, 75(18):5042–5056, 2008.

[63] Milan Jirásek and Peter Grassl. Evaluation of directional mesh bias in concrete fracture simulations using continuum damage models. *Engineering Fracture Mechanics*, 75(8):1921–1943, 2008.

[64] Peter Grassl and Milan Jirásek. Damage-plastic model for concrete failure. *International Journal of Solids and Structures*, 43(22):7166–7196, June 2006.

[65] René de Borst. Fracture in quasi-brittle materials: a review of continuum damage-based approaches. *Engineering Fracture Mechanics*, 69(2):95–112, 2002.

[66] G. Francfort and J.J. Marigo. Vers une théorie énergétique de la rupture fragile. *Comptes Rendus Mécanique*, 330(4):225–234, 2002.

[67] B. Bourdin, GA Francfort, and JJ Marigo. Numerical experiments in revisited brittle fracture. *Journal of the Mechanics and Physics of Solids*, 48(4):797–826, 2000.

[68] M. Charlotte, J. Laverne, and J.J. Marigo. Initiation of cracks with cohesive force models: a variational approach. *European Journal of Mechanics/A Solids*, 25(4):649–669, 2006.

[69] B. Bourdin, G.A. Francfort, and J.J. Marigo. The variational approach to fracture. *Journal of Elasticity*, 91(1):5–148, 2008.

[70] M.E. Gurtin. *Configurational forces as basic concepts of continuum physics*. Springer, 2000.

[71] E. Gürses and C. Miehe. A computational framework of three-dimensional configurational-force-driven brittle crack propagation. *Computer Methods in Applied Mechanics and Engineering*, 198(15–16):1413–1428, 2009.

[72] P. Dumstorff and G. Meschke. Modelling of cohesive and non-cohesive cracks via X-FEM based on global energy criteria. *DR: J. Owen, E. Onate, and B. Suarez, editors, Computational Plasticity*, pages 565–568, 2005.

[73] M. Xie and W.H. Gerstle. Energy-based cohesive crack propagation modeling. *Journal of Engineering Mechanics*, 121:1349, 1995.

[74] T. H. Lin and M. Ito. Theoretical plastic stress-strain relationshop of a polycrystal and the comparisonswith the von mises and the tresca plasticity theories. *Int. J. Eng. Sci.*, 4:543–561, 1966.

[75] A. Paglietti and M. C. Porcu. Stress stability at the yield surface. *Int. J. Non-Linear Mechanics*, 30(2):141–148, 1995.

[76] Jessica Agde Tjernlund, Kristofer Gamstedt, and Peter Gudmundson. Length-scale effects on damage development in tensile loading of glass-sphere filled epoxy. *International Journal of Solids and Structures*, 43(24):7337–7357, May 2006.

[77] R. L. Stratonovitch. Derivation of irreversibility of thermodynamic processes from microscopic reversibility. *Theoretical and Mathematical Physics*, 36(1):607–616, 1978.

[78] Stéphane Bordas, Vinh Phu Nguyen, Cyrille Dunant, Hung Nguyen-Dang, and Amor Guidoum. An extended finite element library. *International Journ Numer Meth. Engng*, 71:703–732, January 2007.

[79] J. C. Simo and T. J. R. Hughes. *Computational Inelasticity*, volume 7. Springer, 1998.

[80] Chengzhi Zhang, Aiqin Wang, Mingshu Tang, Bingqin Wu, and Ningsheng Zhang. Influence of aggregate size and aggregate size grading on asr expansion. *Cement and Concrete Research*, 29:1393 – 1396, 1999.

[81] W. J. French. Avoiding concrete aggregate problems. *Improving Civil Engineering Structures-Old and New. Geotechnical Publishing Ltd*, pages 65–95, 1994.

[82] Per Arne Dahl, Jan Lindgård, Svein Willy Danielsen, Cecilie Hagby, Reidar Kompen, Bård Pedersen, and Terje F. Rønning. Specifications and Guidelines for Production of AAR Resistant Concrete in Norway. In *12th ICAAR*, pages xxx – xxx, October 2004.

[83] D. W. Hobbs. Aggregate influence on chloride ion diffusion into concrete. *Cement and Concrete Research*, 29(12):1995 – 1998, 1999.

[84] B. J. Wigum, W. J. French, R. J. Howarth, and C. Hills. Accelerated tests for assessing the potential exhibited by concrete aggregates for alkali-aggregate reaction. *Cement and Concrete Composites*, 19(5):451—476, 1997.

[85] B. J. Wigum. *Alkali-aggregate reactions in concrete: properties, classification and testing of Norwegian cataclastic rocks*, chapter Properties of Alkali-Reactive Aggregates, pages 22–43. University of Trondheim, 1995.

[86] Isabelle Comby-Peyrot. *Development And Validation Of A 3D Computational Tool To Describe Damage And Fracture Due To Alkali Silica Reaction In Concrete Structures.* PhD thesis, Mines ParisTech, Décembre 2006.

[87] Zdeněk P. Bažant and Alexander Steffens. Mathematical model for kinetics of alkali-silica reaction in concrete. *Cement and Concrete Research*, 30(3):419–428, December 1999.

[88] Martin Cyr, Patrice Rivard, and Francis Labrecque. Reduction of asr-expansion using powders ground from various sources of reactive aggregates. *Cement & Concrete Composites*, xxx:xxx – xxx, 2009.

[89] K. Ramyar, A. Topal, and Ö. Andiç. Effects of aggregate size and angularity on alkali–silica reaction. *Cement and Concrete Research*, 35(11):2165–2169, 2005.

[90] B. P. Hughes and J. E. Ash. Anisotropy and failure criteria for concrete. *Materials and Structures*, 3(6):371–374, 1970.

[91] Alexander Steffens, K. Li, and O. Coussy. Aging Approach to Water Effect on Alkali–Silica Reaction Degradation of Structures. *Journal of Engineering Mechanics*, 129:50, 2003.

[92] F. Bangert, D. Kuhl, and G. Meschke. Chemo-hygro-mechanical modelling and numerical simulation of concrete deterioration caused by alkali-silica reaction. *International Journal for Numerical and Analytical Methods in Geomechanics*, 28(7–8):689–714, 2004.

[93] E. Grimal, A. Sellier, Y. Le Pape, and E. Bourdarot. Creep, Shrinkage, and Anisotropic Damage in Alkali-Aggregate Reaction Swelling Mechanism-Part I: A Constitutive Model. *ACI Materials Journal*, 105(3), 2008.

[94] A. Winnicki and S. Pietruszczak. On Mechanical Degradation of Reinforced Concrete Affected by Alkali-Silica Reaction. *Journal of Engineering Mechanics*, 134:611, 2008.

[95] B. Capra and J.-P. Bournazel. Modeling of induced mechanical effects of alkali-aggregate reactions. *Cement and Concrete Research*, 28(2):251–260, 1998.

[96] Manuel F. Herrador, Fernando Martínez-Abella, and Juan Ramón Rabuñal Dopico. Experimental evaluation of expansive behavior of an old-aged ASR-affected dam concrete: methodology and application. *Materials and Strucures*, 41:173–188, 2008.

[97] C. F. Ferraris, E. J. Garboczi, F. L. Davis, and J. R. Clifton. The effect of stress relaxation, self-desiccation, and water absorption on the alkali-silica reaction in low water/cement ratio mortars. *Cement and Concrete Research*, 27(10):1553–1560, 1997.

[98] Théodore Chappex. ASR in the Serra Dam. Master's thesis, École Polytechnique Fédérale de Lausanne, 2008.

[99] AFNOR. Méthodes d'essai de réactivité aux alcalis, 2 2004.

[100] J. Bolomey. Granulation et prévision de la résistance probable des bétons. *Travaux*, 30(19):228–232, 1935.

[101] I. Sims and P. Nixon. RILEM Recommended Test Method AAR-0: Detection of Alkali-Reactivity Potential in Concrete—Outline guide to the use of RILEM methods in assessments of aggregates for potential alkali-reactivity. *Materials and Structures*, 36(7):472–479, 2003.

[102] J. Sklansky. Image segmentation and feature extraction. *IEEE Transactions on Systems, Man and Cybernetics*, 8(4):237–247, 1978.

[103] Xinyu Zhang. *Quantitative microstructural characterisation of concrete cured under realistic temperature conditions.* PhD thesis, École Polytechnique Fédérale de Lausanne, 2007.

[104] T. C. Powers. Structure and physical properties of hardened Portland cement paste. *Journal of the American Ceramic Society*, 41(1):1–6, 1958.

[105] Eduardo M. R. Fairbarn, Fernando L. B. Ribeiro, Luciana E. Lopes, Romildo D. Toledo-Filho, and Marcos M. Silvoso. Modelling the structural behaviour of a dam affected by alkali–silica reaction. *Commun. Numer. Meth. Engng*, 22(1):1–12, July 2005.

[106] Tsuneki Ichikawa and Masazumi Miura. Modified model of alkali-silica reaction. *Cement and Concrete Research*, 37(9):1291–1297, June 2007.

[107] Oğuzhan Çopuroğlu and Erik Schlangen. Modelling of effect of asr on concrete microstructure. *Key Engineering Materials*, 348:809–812, 09 2007.

[108] Eric Garcia-Diaz, J. Riche, D. Bulteel, and C. Vernet. Mechanism of damage for the alkali–silica reaction. *Cement and Concrete Research*, 36(2):395–400, June 2006.

[109] I. Babuška and I. Melenk. Partition of unity method. *International Journ Numer Meth. Engng*, 40(4):727–758, 1997.

[110] J. Dolbow, E. Fried, and H. Ji. Chemically induced swelling of hydrogels. *Journal of the Mechanics and Physics of Solids*, 52(1):51–84, 2004.

[111] Georgios Constantinides and Franz-Josef Ulm. The effect of two types of C-S-H term on the elasticity of cement-based materials: Results from nanoindentation and micromechanical modeling. *Cement and Concrete Research*, 34(1):67–80, 2004.

[112] Günter Bach. Über die Größenverteilung von Kugelschnitten in durchsichtigen Schnitten endlicher Dicke. *Zeitschrift für Angewandte Mathematik und Mechanik*, 38(7–8):256–258, 1959.

[113] P. L. Goldsmith. The calculation of true particle size distributions from the sizes observed in a thin slice. *British Journal of Applied Physics*, 18(6):813–830, 1967.

[114] S. S. Muchnick. *Advanced compiler design and implementation.* Morgan Kaufmann, 1997.

[115] Andrew W. Appel. Ssa is functional programming. *SIGPLAN Not.*, 33(4):17–20, 1998.

[116] J. Cocke. Global common subexpression elimination. In *Proceedings of a symposium on Compiler Optimization*, volume 5, pages 20–24. ACM New York, NY, USA, 1970.

[117] J.R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. *Unpublished paper.*

[118] S. Bishnoi and K.L. Scrivener. μic: A new platform for modelling the hydration of cements. *Cement and Concrete Research*, 39(4):266–274, 2009.

[119] Valeria La Saponara and George A. Kardomateas. Crack branching in cross-ply composites: an experimental study. *Composite Structures*, 53:333–344, 2001.

[120] K. Y. Lam, P. P. Ong, and N. Wude. Interaction between a circular inclusion and a symmetrically branched crack. *Theoretical and applied fracture mechanics*, 28:197–211, 1998.

[121] Tao Wang, Jun Luo, Zhongmin Xiao, and Jianqiao Chen. On the nucleation of a zener crack from a wedge disinclination dipole in the presence of a circular inhomogeneity. *European Journal of Mechanics A/Solids*, 28:688–696, 2009.

[122] P.K. Mehta and Paulo J. M. Monteiro. *Concrete: microstructure, properties, and materials*. McGraw-Hill, 2006.

[123] Karen L. Scrivener and E. M. Gartner. Microstructure gradient in cement paste around aggregate particles. In S. Mindess and S. P. Shah, editors, *Bonding in Cementitious Composites*, pages 77–85. Materials Research Society, 1988.

[124] Karen L. Scrivener and Kamran M. Nemati. The percolation of pore space in the cement paste/aggregate interfacial zone of concrete. *Cement and Concrete Research*, 26(1):35–40, 1996.

[125] Lucie Baillon. *Application des Techniques d'émissions acoustiques au suivi de la fissuration dans le béton*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2002.

[126] Manu Santhanam, Menashi D. Cohen, and Jan Olek. Mechanism of sulfate attack: A fresh look: Part 1: Summary of experimental results. *Cement and Concrete Research*, 32(6):915–921, 2002.

[127] Manu Santhanam, Menashi D. Cohen, and Jan Olek. Mechanism of sulfate attack a fresh look: Part 2. proposed mechanisms. *Cement and Concrete Research*, 33(3):341–346, 2003.

List of Acronyms

sk	Swiss non-reactive aggregate SK
sa	Swiss reactive aggregate SA
sf	Swiss reactive aggregate SF
si	Swiss reactive aggregate SI
ss	Swiss reactive aggregate SS
aj	American reactive aggregate AJ
ai	American reactive aggregate AR
al	American reactive aggregate AL
aw	American reactive aggregate AW
as	American reactive aggregate AS
afnor	Agence Française de Normalisation
amie	Automated meshing for integrated experiments
aomd	algorithm oriented mesh database
api	application programming interface
aar	alkali-aggregate reaction
asr	alkali-silica reaction
astm	American Society of Testing and Materials
bse	back-scattered electron
cad	computer-aided design
C-S-H	calcium silicate hydrate
cse	common sub-expression elimination
$\mathbf{D_{max}}$	maximum diameter
fem	finite element modeling
fortran	the formula translating system
gui	graphical user interface
ia	image analysis
io	input-output
lapack	the linear algebra package
lmc	Laboratory of Construction Materials
mls	moving least square
$\mathbf{Na_2O_{eq}}$	sodium oxide equivalent
nurbs	non-uniform rational B-spline

petsc	the portable, extensible toolkit for scientific computation
psd	particle size distribution
rev	representative elementary volume
sem	scanning electron microscopy
sor	successive over-relaxation
ssa	static single assignment
vm	virtual machine
$\frac{w}{c}$	water to cement ratio
xfem	extended finite element modeling
xrd	X-Ray diffraction

Index

APPENDIX A

Image Segmentation Algorithm

Presented here is the complete code for the image segmentation algorithm. The dependencies are Qt4 and OpenCV.

```
#include <fstream>
#include <valarray>
#include <iostream>
#include <string>
#include <vector>
#include <complex>
#include <map>
#include <fstream>

#include <QImage>
#include <QImageWriter>
#include <QStringList>
#include <QFileDialog>
#include <QApplication>

#include <opencv/cv.h>
#include <opencv/cxtypes.h>
#include <opencv/highgui.h>
#include <opencv/cvaux.h>
#include <opencv/cxmisc.h>

struct SegmentationData
{
  SegmentationData(): aggregateCount(0), damageCount(0),
      gelCount(0) { }
  std::vector<std::pair<int, int> > threshold ;
  size_t aggregateCount ;
  std::vector<size_t > damageCount ;
  std::vector<size_t> gelCount ;
} ;

class BMPFile
{
  std::valarray<quint8> data ;
  std::valarray<quint8> sortedData ;
  unsigned int rows ;
  unsigned int cols ;
  unsigned int numvals ;
public:
  BMPFile(QString filename)
  {
    rows = 0 ;
    cols = 0 ;
    numvals = 0 ;
    data.resize(0) ;
    QImage image(filename) ;

    rows = image.height() ;
    cols = image.width() ;
    numvals = cols*rows ;
    data.resize(numvals) ;

    for(size_t i = 0 ; i < rows ; i++)
    {
      for(size_t j = 0 ; j < cols ; j++)
      {
        data[i*cols+j] = qGray(image.pixel(j,i)) ;
      }
    }
    sortedData.resize(numvals) ;
    sortedData = data ;
    std::stable_sort(&sortedData[0], &sortedData[numvals]) ;
  }

    std::valarray<quint8> & getData()
  {
    return data ;
  }

  const std::valarray<quint8> & getData() const
  {
    return data ;
  }

  std::valarray<quint8> & getSortedData()
  {
    return sortedData ;
  }

  const std::valarray<quint8> & getSortedData() const
  {
    return sortedData ;
```

```
}

unsigned int getRows() const
{
  return rows ;
}

unsigned int getCols() const
{
  return cols ;
}

unsigned int getNumberOfNonZeroValues() const
{
  return numvals ;
}

std::valarray<size_t> getHistogram() const
{
  std::valarray<size_t> ret((size_t)0, (size_t)256) ;

  quint8 current = sortedData[0] ;

  int f = 0 ;
  for(size_t i = 0 ; i < numvals ; i++)
  {
    while(i < numvals && sortedData[i] == current)
    {
      f++ ;
      i++ ;
    }

    ret[current] = f ;
    f= 0 ;
    if( i < numvals)
      current = sortedData[i] ;
  }

  return ret ;
}
//post-analysis on segmented image
void filterLargeDamage(QString filename, int crackGrey = 160,
    int gelGrey = 80)
{
  QImage img(filename) ;

  std::valarray<quint8> aggregateArray(numvals) ;
  std::valarray<quint8> gelArray(numvals) ;

  for(size_t i = 0 ; i < rows ; i++)
  {
    for(size_t j = 0 ; j < cols ; j++)
    {
        aggregateArray[i*cols+j] = (qGray(img.pixel(j,i)) > 1)
            *255;
        gelArray[i*cols+j] = (qGray(img.pixel(j,i)) > 1 && qGray
            (img.pixel(j,i)) <= gelGrey)*255;
    }
}

CvMat aggMask ;
CvMat gelMask ;

cvInitMatHeader(&aggMask, rows, cols, CV_8UC1, &
    aggregateArray[0]) ;
cvInitMatHeader(&gelMask, rows, cols, CV_8UC1, & gelArray
    [0]) ;


CvMemStorage * storage = cvCreateMemStorage(0);
CvContour * contour = NULL;

cvFindContours( &aggMask, storage, (CvSeq **)&contour,
    sizeof(CvContour), CV_RETR_EXTERNAL,
    CV_CHAIN_APPROX_SIMPLE );

std::vector<double> aggAreas ;
std::vector<double> gelAreas ;
//liste des aires des aggregats
for( ; contour != NULL; contour = (CvContour*)contour->
    h_next )
{
  aggAreas.push_back(fabs(cvContourArea(contour))) ;
}

contour = NULL;

cvFindContours( &gelMask, storage, (CvSeq **)&contour,
    sizeof(CvContour), CV_RETR_EXTERNAL,
    CV_CHAIN_APPROX_SIMPLE );

//list gel contour areas
for( ; contour != NULL; contour = (CvContour*)contour->
    h_next )
{
  gelAreas.push_back(fabs(cvContourArea(contour))) ;
}

contour = NULL;

cvReleaseMemStorage(&storage) ;
//sort the aggregates according to their area size
std::sort(aggAreas.begin(), aggAreas.end()) ;
std::sort(gelAreas.begin(), gelAreas.end()) ;

std::vector<double> cumulativeAgg( 100) ;
std::vector<double> cumulativeGel( 100) ;
```

```
    double totAgg = 1;
    double totGel = 1;

    double maxAgg = *aggAreas.rbegin() ;
    double minAgg = *aggAreas.begin() ;
    double maxGel = *gelAreas.rbegin() ;
    double minGel = *gelAreas.begin() ;

    std::cout << "we␣have␣" << aggAreas.size() << "␣aggregates"
        << std::endl ;
    for(size_t i = 0 ; i < aggAreas.size() ; i++)
    {
      cumulativeAgg[(double)((sqrt(aggAreas[i])-sqrt(minAgg))/(
          sqrt(maxAgg)-sqrt(minAgg)))*99] += aggAreas[i] ;
      totAgg += aggAreas[i];
    }

    std::cout << "we␣have␣" << gelAreas.size() << "␣gel␣zones"
        << std::endl ;

    for(size_t i = 0 ; i < gelAreas.size() ; i++)
    {
      cumulativeGel[(double)(gelAreas[i]-minGel)/(maxGel-minGel)
          *99] += gelAreas[i] ;
      totGel += gelAreas[i];
    }

    std::cout << "␣aggregate␣" << std::endl ;
    for(size_t i = 0 ; i < 100 ; i++)
      std::cout << (double)(cumulativeAgg[i]/totAgg*100.) << std
          ::endl ;

    std::cout << "␣gel␣" << std::endl ;
    for(size_t i = 0 ; i < 100 ; i++)
      std::cout << (double)(cumulativeGel[i]/totGel*100.) <<
          std::endl ;
    //excessively large gel pockets
    std::cout << "last␣centile␣of␣degradation␣:␣" << (double)(
        cumulativeGel[99]/totGel*100.) << "\%␣of␣total␣
        degradation" << std::endl ;
  }

  void segment(std::valarray<quint8> * src, std::vector<CvMat *>
      & mask,  SegmentationData & results, int downval, int
      upval, int areaMin = 200) const
  {
    std::cout << "segmenting..." << std::endl ;
    results.threshold.push_back(std::make_pair(downval, upval))
        ;
    for(size_t i = 0 ; i < numvals ; i++)
    {
      if((*src)[i] > results.threshold.back().first && (*src)[i]
          < results.threshold.back().second)
        (*src)[i] = 255 ;
      else
        (*src)[i] = 0 ;
    }
    mask.push_back(cvCreateMatHeader(rows, cols, CV_8UC1)) ;
    cvSetData(mask.back(), (void *)&(*src)[0], cols) ;
    CvMemStorage * storage = cvCreateMemStorage(0);
    CvContour * contour = NULL;
//
    cvFindContours( mask.back(), storage, (CvSeq **)&contour,
        sizeof(CvContour), CV_RETR_EXTERNAL,
        CV_CHAIN_APPROX_SIMPLE );

    for( ; contour != NULL; contour = (CvContour*)contour->
        h_next )
    {
      double area = fabs(cvContourArea(contour)) ;
      if (area > areaMin )
      {
        cvDrawContours( mask.back(),
                        (CvSeq *)contour,
                        CV_RGB(255,255,255),
                        CV_RGB(255,255,255),
                        -1, CV_FILLED, 8 );
      }
      else
      {
        cvDrawContours( mask.back(),
                        (CvSeq *)contour,
                        CV_RGB(0,0,0),
                        CV_RGB(0,0,0),
                        -1, CV_FILLED, 8 );
      }
    }
    contour = NULL;
    cvReleaseMemStorage( &storage );
  }

  void writePng(QString filename, const float fraction, std::
      vector<SegmentationData> & pixelCount, const std::vector<
      std::pair<int, int> > thresholds, int crackGreyThreshold =
      50) const
  {
    if(thresholds.empty())
      return ;
    QImage img(filename) ;
    QImage image( cols, rows, QImage::Format_Indexed8 ) ;
    image.setNumColors ( 256 ) ;
    for(size_t i = 0 ; i < 256 ; i++)
    {
      QRgb value;
      value = qRgb(i, i, i);
       image.setColor(i, value);
    }
```

```
SegmentationData results ;
std::valarray<quint8> *charmask = new std::valarray<quint8>(
    numvals) ;
std::valarray<quint8> *charmasknocracks = new std::valarray<
    quint8>(numvals) ;
int datamax = data.max() ;
int datamin = data.min() ;
for(size_t i = 0 ; i < numvals ; i++)
{
  (*charmask)[i] = (quint8)(data[i]) ;
}

IplConvKernel* structureElement =
    cvCreateStructuringElementEx( 5, 5, 3, 3, CV_SHAPE_CROSS
    );

std::vector<CvMat *> mask ;
std::vector<std::valarray<quint8> *> charmaskphase ;
CvMat temp ;

charmaskphase.push_back(new std::valarray<quint8>((quint8)0,
     charmask->size()));
CvMat * maskarray = cvCreateMatHeader(rows, cols, CV_8UC1);
cvSetData(maskarray, (void *)&(*(charmaskphase.back()))[0],
    cols) ;
charmaskphase.push_back(new std::valarray<quint8>((quint8)0,
     charmask->size()));
CvMat * pasteMask = cvCreateMatHeader(rows, cols, CV_8UC1);
cvSetData(pasteMask, (void *)&(*(charmaskphase.back()))[0],
    cols) ;

//grey bands for the various aggregate phases
for(size_t i = 0 ; i < thresholds.size() ; i++)
{
  charmaskphase.push_back(new std::valarray<quint8>((*
      charmask)));
  segment(charmaskphase.back(),mask,results,thresholds[i].
      first,thresholds[i].second);
  cvMax( maskarray, mask[i], maskarray);
  cvReleaseData(mask[i] );
  cvReleaseMat(&mask[i] );
}


cvMorphologyEx( maskarray , maskarray, &temp,
    structureElement, CV_MOP_CLOSE, 2 );
cvThreshold(maskarray, pasteMask, 100, 255,
    CV_THRESH_BINARY_INV) ;

CvContour * contour = NULL;
CvMemStorage * storage = cvCreateMemStorage(0);
```

```
cvFindContours( pasteMask, storage, (CvSeq **)&contour,
    sizeof(CvContour), CV_RETR_CCOMP, CV_CHAIN_APPROX_SIMPLE
    );

CvContour * largest = contour;
double maxArea = 0 ;
for( ; contour != NULL; contour = (CvContour*)contour->
    h_next )
{
  double area = fabs(cvContourArea(contour)) ;
  if (area > 750 )
  {
    cvDrawContours( pasteMask,
      (CvSeq *)contour,
      CV_RGB(255,255,255),
      CV_RGB(255,255,255),
      -1, CV_FILLED, 8 );
  }
  else
  {
    cvDrawContours( pasteMask,
      (CvSeq *)contour,
      CV_RGB(0,0,0),
      CV_RGB(0,0,0),
      -1, CV_FILLED, 8 );
  }
}

contour = NULL;
cvMorphologyEx( pasteMask , pasteMask, &temp, NULL,
    CV_MOP_CLOSE, 2 );
cvThreshold(pasteMask, maskarray, 100, 255,
    CV_THRESH_BINARY_INV) ;

cvFindContours( maskarray, storage, (CvSeq **)&contour,
    sizeof(CvContour), CV_RETR_EXTERNAL,
    CV_CHAIN_APPROX_SIMPLE );

for( ; contour != NULL; contour = (CvContour*)contour->
    h_next )
{
  double area = fabs(cvContourArea(contour)) ;
  double length = cvArcLength( contour );


  CvRect rec = cvBoundingRect(contour);
  double recArea = rec.width*rec.height ;

  double formFactor = area/recArea ;

  if ((area < 3000 && formFactor < 0.55) || (area < 1000)
      || (formFactor < .60 && area < 5000))
  {
    cvDrawContours( maskarray,
          (CvSeq *)contour,
          CV_RGB(0,0,0),
       CV_RGB(0,0,0),
          -1, CV_FILLED, 8 );
    }
    else
    {
        cvDrawContours( maskarray,
          (CvSeq *)contour,
          CV_RGB(255,255,255),
          CV_RGB(255,255,255),
          -1, CV_FILLED, 8 );
    }
  }

  cvMorphologyEx( maskarray , pasteMask, &temp, NULL,
      CV_MOP_CLOSE, 1 );

  std::copy(pasteMask->data.ptr, pasteMask->data.ptr+numvals,
      &(*charmask)[0]) ;
  cvReleaseData(maskarray);
  cvReleaseMat(&maskarray);
  cvReleaseData(pasteMask);
  cvReleaseMat(&pasteMask);

  cvReleaseMemStorage(&storage) ;
  cvReleaseStructuringElement(&structureElement) ;

  double aggCount = 0 ;
  double crackCount = 0 ;
  double damageCount = 0 ;

  std::valarray<size_t> variableCrackCount(size_t(0), 40) ;
  std::valarray<size_t> variableDamageCount(size_t(0), 40) ;

  //testing various possible crack grey levels
  for(size_t i = 0 ; i < numvals ; i++)
  {
    (*charmasknocracks)[i] = 0 ;
    if((*charmask)[i] && data[i] < crackGreyThreshold)
    {
      damageCount++ ;
      aggCount++ ;
      (*charmask)[i] = 160 ;
      (*charmasknocracks)[i] = 255 ;
    }
    else if((*charmask)[i])
      aggCount++ ;

  }
  CvMat * crackmask = cvCreateMatHeader(rows, cols, CV_8UC1);
  charmaskphase.push_back(new std::valarray<quint8>((*
      charmasknocracks)));
```

```
cvSetData(crackmask, (quint8 *)&(*charmaskphase.back())[0],
    cols) ;
cvMorphologyEx( crackmask , crackmask, &temp, NULL,
    CV_MOP_OPEN, 1 );

std::copy(crackmask->data.ptr, crackmask->data.ptr+numvals,
    &(*charmasknocracks)[0]) ;

for(size_t i = 0 ; i < numvals ; i++)
{
  if((*charmasknocracks)[i])
  {
    crackCount++ ;
    (*charmask)[i] = 80 ;
  }
}

//crack grey levels
for(size_t t = std::max(crackGreyThreshold-20, 0) ; t < std
    ::max(crackGreyThreshold-20, 0)+40 ; t++)
{
  for(size_t i = 0 ; i < numvals ; i++)
  {
    (*charmasknocracks)[i] = 0 ;
    if((*charmask)[i] && data[i] < t)
    {
      (*charmasknocracks)[i] = 255 ;
      variableCrackCount[t - std::max(crackGreyThreshold-20,
          0)]++ ;
    }

  }

  CvMat * crackmaskvar = cvCreateMatHeader(rows, cols,
      CV_8UC1);
  charmaskphase.push_back(new std::valarray<quint8>((*
      charmasknocracks)));
  cvSetData(crackmaskvar, (quint8 *)&(*charmaskphase.back())
      [0],cols) ;
  cvMorphologyEx( crackmaskvar , crackmaskvar, &temp, NULL,
      CV_MOP_OPEN, 1 );

  std::copy(crackmaskvar->data.ptr, crackmaskvar->data.ptr+
      numvals, &(*charmasknocracks)[0]) ;
  for(size_t i = 0 ; i < numvals ; i++)
  {
    if((*charmasknocracks)[i])
      variableDamageCount[t-std::max(crackGreyThreshold-20,
          0)]++ ;
  }
  cvReleaseData(crackmaskvar );
  cvReleaseMat(&crackmaskvar );
}
```

```
    cvReleaseMat(&crackmask ) ;

    results.aggregateCount = aggCount ;
    for(size_t i = 0 ; i < variableDamageCount.size() ; i++)
      results.damageCount.push_back(variableDamageCount[i]);
    for(size_t i = 0 ; i < variableCrackCount.size() ; i++)
      results.gelCount.push_back(variableCrackCount[i]);

    pixelCount.push_back(results) ;
    std::cout << damageCount<< "␣␣␣␣"<< crackCount << "␣␣␣" <<
        aggCount << std::endl ;

    int idx = 0 ;
    for(size_t i = 0 ; i < rows ; i++)
    {
      for(size_t j = 0 ; j < cols ; j++)
      {
        image.setPixel(j, i,(quint8)(*charmask)[idx]) ;
        idx++ ;
      }
    }

    QString name("") ;
    name += filename ;
    name += QString("MODE.png") ;
    QImageWriter writer(name, "png");
    writer.setQuality(0) ; //means highest compression
    writer.write(image);

    for(size_t i = 0 ; i < charmaskphase.size() ; i++)
      delete charmaskphase[i] ;

    delete charmask ;
    delete charmasknocracks ;
    cvReleaseStructuringElement(&structureElement) ;
  }

  void PrintHistogram() const
  {
    int f = 0 ;

    quint8 current = sortedData[0] ;

    for(size_t i = 0 ; i < numvals ; i++)
    {
      while(i < numvals && sortedData[i] == current)
      {
        f++ ;
        i++ ;
      }

      std::cout << (int)current << "␣␣␣" << f << std::endl ;
      f= 0 ;
      if( i < numvals)
```

```
        current = sortedData[i] ;
    }
  }
} ;

void crackDistribution(QString source, int threshold)
{
  QImage image(source) ;
  QImage mask(source+"mode.png") ;

  int rows = image.height() ;
  int cols = image.width() ;
  int numvals = cols*rows ;
  std::valarray<quint8> data(numvals) ;

  for(size_t i = 0 ; i < rows ; i++)
  {
    for(size_t j = 0 ; j < cols ; j++)
    {
      data[i*cols+j] = (qGray(image.pixel(j,i)) <= threshold &&
          qGray(mask.pixel(j,i)) > 0) ;
    }
  }

  CvMat cracks ;
  cvInitMatHeader(&cracks, rows, cols, CV_8UC1, &data[0]) ;
  size_t tot = cvSum(&cracks).val[0] ;

  std::cout << tot << std::endl ;
  while(tot > 0)
  {
    cvErode( &cracks , &cracks, NULL, 1);
    tot = cvSum(&cracks).val[0] ;
    std::cout << tot << std::endl ;
  }
}

int main(int argc, char *argv[]) {
  QApplication app(argc, argv);

  bool printHistogram = false ;
  bool doAnalysis = true ;
  bool printCumulativeFreqs = false ;
  bool crackdistribution = false ;
  bool setoutput = false ;
  int crackGreyThreshold = 50 ;
  std::vector<std::pair<int, int> > thresholds ;
  std::string outfilename("/Users/lionelsofia/Desktop/
      pixel_count.txt") ;
  QString source;
  int threshold  = 80;
  // number of tested grey thresholds for the cracks
  int greys = 40 ;
```

```
for(size_t i = 1 ; i < argc ; i++)
{
  if(std::string(argv[i]) == std::string("--help"))
  {
    std::cout << "usage:" << std::endl ;
    std::cout << "--histogram␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣print␣global␣
        histogram" << std::endl ;
    std::cout << "--no-analysis␣␣␣␣␣␣␣␣␣␣␣␣␣␣do␣not␣perform␣
        analysis" << std::endl ;
    std::cout << "--cumulative-frequencies␣␣␣print␣cumulative␣
        freqs␣for␣gel␣and␣aggs" << std::endl ;
    std::cout << "--crack-distribution␣␣<file.bmp>␣<threshold>
        ␣␣␣␣␣print␣crack␣distribution" << std::endl ;
    std::cout << "--threshold␣<down>␣<up>␣␣␣␣set␣a␣threshold"
        << std::endl ;
    std::cout << "--crack-threshold␣<up>␣␣␣␣␣set␣crack␣grey␣
        threshold" << std::endl ;
    std::cout << "--outfile␣<results.txt>␣␣␣␣sets␣the␣output␣
        file" << std::endl ;
    return 0 ;
  }
  if(std::string(argv[i]) == std::string("--histogram"))
    printHistogram = true ;
  if(std::string(argv[i]) == std::string("--no-analysis"))
    doAnalysis = false ;
  if(std::string(argv[i]) == std::string("--cumulative-
      frequencies"))
    printCumulativeFreqs = true ;
  if(std::string(argv[i]) == std::string("--crack-distribution
      "))
  {
    source = argv[i+1] ;
    threshold = atoi(argv[i+2]) ;
    crackDistribution(source, threshold) ;
    return 0 ;
  }
  if(std::string(argv[i]) == std::string("--crack-threshold"))
  {
    crackGreyThreshold = atoi(argv[i+1]) ;
    i++ ;
  }
  if(std::string(argv[i]) == std::string("--threshold"))
  {
    thresholds.push_back(std::make_pair(atoi(argv[i+1]),atoi(
        argv[i+2]))) ;
    i+=2 ;
  }
  if(std::string(argv[i]) == std::string("--outfile"))
  {
    outfilename = std::string(argv[i+1]) ;
    setoutput = true ;
    i++ ;
  }
}
```

```
QString dirname  = QFileDialog::getExistingDirectory(0, "Open␣
    Directory", ".",QFileDialog::ShowDirsOnly | QFileDialog::
    DontResolveSymlinks);
QDir directory(dirname);
QStringList filters;
filters << "*.BMP" << "*.bmp" ;
directory.setNameFilters(filters);
QStringList files = directory.entryList(QDir::Files | QDir::
    NoDotAndDotDot | QDir::Readable) ;

std::vector<SegmentationData > pixelCount ;

std::vector<double> averageDamage(greys) ;
std::vector<double> averageReaction(greys) ;
std::vector<double> averageTotal(greys) ;

std::valarray<size_t > histogram((size_t)0, (size_t)256) ;

if(printHistogram)
{
  for (QStringList::const_iterator constIterator = files.
      constBegin();
  constIterator != files.constEnd();
  ++constIterator)
  {
      QString absoluteFileName (dirname + "/"+ *constIterator)
          ;

      std::cout << absoluteFileName.toLocal8Bit().constData()
          << std::endl ;
      BMPFile file(absoluteFileName);
      histogram +=file.getHistogram() ;
    }

    for(size_t i = 0 ; i < 256 ; i++)
    {
      std::cout << histogram[i] << std::endl ;
    }
  }

if(printCumulativeFreqs)
{
  for (QStringList::const_iterator constIterator = files.
      constBegin();
  constIterator != files.constEnd();
  ++constIterator)
  {
    QString absoluteFileName (dirname + "/"+ *constIterator);
    QString fileName (dirname + "/"+ *constIterator + "MODE.
        png");
    std::cout << absoluteFileName.toLocal8Bit().constData() <<
         std::endl ;
      BMPFile file(absoluteFileName);
      file.filterLargeDamage(fileName) ;
    }
  }

  bool haveFiles = false ;
  if(doAnalysis)
  {
    std::ofstream outfile(outfilename.c_str()) ;
    std::map<double, SegmentationData> sortedData ;
//  int filecount = 0 ;
    for (QStringList::const_iterator constIterator = files.
        constBegin();
      constIterator != files.constEnd();
      ++constIterator)
    {
        haveFiles=true ;

        QString absoluteFileName (dirname + "/"+ *constIterator)
            ;
        std::cout << absoluteFileName.toLocal8Bit().constData()
            << std::endl ;
        BMPFile file(absoluteFileName);
        if(thresholds.empty())
          return 0 ;

        file.writePng(absoluteFileName, .9995, pixelCount,
            thresholds, crackGreyThreshold) ;
        //the first value is the aggregate count
        double vtot = pixelCount.rbegin()->aggregateCount ;
        double numFiles = files.size() ;

        outfile << constIterator->toLocal8Bit().constData() << "
            \t" << pixelCount.rbegin()->aggregateCount << std::
            endl ;
        outfile << "Damage\t" ;
        //those are the values after erosion = gel
        for(size_t i = 0 ; i < pixelCount.rbegin()->damageCount.
            size() ; i++)
        {
          outfile << "\t"<<pixelCount.rbegin()->damageCount[i] ;
          if(vtot)
          {
            double reaction = (double)pixelCount.rbegin()->
                damageCount[i]/vtot ;
            sortedData[reaction] = *pixelCount.rbegin() ;
            averageReaction[i] += reaction/numFiles ;
          }
        }
        outfile << std::endl ;
        outfile << "Reaction\t" ;
        //those are the values before erosion = gel+ cracks
        for(size_t i = 0 ; i < pixelCount.rbegin()->gelCount.
            size() ; i++)
        {
          outfile << "\t"<< pixelCount.rbegin()->gelCount[i] ;
          if(vtot)
          {
            averageTotal[i] += ((double)pixelCount.rbegin()->
                gelCount[i]/vtot)/numFiles ;
          }
        }
        for(size_t i = 0 ; i < averageDamage.size() ; i++)
        {
          if(vtot)
          {
          //so the cracks are the total-gel
            averageDamage[i] = (averageTotal[i]-averageReaction[
                i]);
          }
        }

        outfile << std::endl ;
    }

    if(!haveFiles)
    {
      std::cout << "no␣files␣!" << std::endl ;
      return 0 ;
    }
    outfile << "niveau␣de␣gris" << std::endl ;
    outfile  << std::endl ;
    if(!sortedData.empty())
    {
      for(int i = 0 ; i < sortedData.begin()->second.threshold.
          size() ; i++)
        outfile << sortedData.begin()->second.threshold[i].first
             << "\t" << sortedData.begin()->second.threshold[i].
            second << std::endl ;
    }

    outfile << "total␣\t␣part␣cracked␣\t␣part␣gel" << std::endl
        ;
    outfile  << std::endl ;
    for(size_t i = 0 ; i < averageDamage.size() ; i++)
    {
      outfile << averageTotal[i] << "\t"<< averageDamage[i] << "
          \t"<< averageReaction[i] << "\t" << std::endl ;
    }
    outfile << "total␣(90<%)␣\t␣part␣cracked␣(90\%)␣\t␣part␣gel␣
        (90\%)" << std::endl ;
    outfile  << std::endl ;
    size_t fivePercent = sortedData.size()/20 ;
    double count = 0 ;
    std::vector<double> averageDamage90(greys) ;
    std::vector<double> averageReaction90(greys) ;
    std::vector<double> averageTotal90(greys) ;
    std::vector<SegmentationData> sortedValues ;
    for(std::map<double, SegmentationData >::iterator i =
        sortedData.begin() ;i!= sortedData.end() ;++i)
    {
      sortedValues.push_back(i->second) ;
      count++ ;
    }

    for(std::vector<SegmentationData >::iterator i = sortedValues
        .begin()+fivePercent ; i != sortedValues.end()-
        fivePercent ;++i)
    {
      //the first value is the aggregate count
      double vtot = i->aggregateCount ;

      //those are the values after erosion = gel
      for(size_t j = 0 ; j < i->damageCount.size() ; j++)
      {
        if(vtot)
        {
          double reaction = (double)i->damageCount[j]/vtot ;
          averageReaction90[j] += reaction/count ;
        }
      }

      //those are the values before erosion = gel+ cracks
      for(size_t j = 0 ; j < i->gelCount.size() ; j++)
      {
        if(vtot)
        {
          averageTotal90[j] += (double)i->gelCount[j]/vtot/count
              ;
        }
      }
      for(size_t j = 0 ; j < averageDamage.size() ;j++)
      {
        if(vtot)
        {
        //so the cracks are the total-gel
          averageDamage90[j] = (averageTotal90[j]-
              averageReaction90[j]);
        }
      }
    }

    for(size_t i = 0 ; i < averageDamage.size() ; i++)
    {
      outfile << averageTotal90[i] << "\t"<< averageDamage90[i]
          << "\t"<< averageReaction90[i] << "\t" << std::endl ;
    }
    outfile.close() ;
  }
  return 0 ;
}
```

APPENDIX B

Numerical Implementation of the Galerkin Method using a Virtual Machine

B.1 Overview

The code should be a representation of the analytical method. It should therefore define the same entities, and be expressed in a way that is close. Ideally, it should not introduce new limitations. The entities used to express the Galerkin method are:

1. Functions
2. Differential operators
3. An integral operator
4. A linear system of equations (implicitly, a way to solve it)

Automated meshing for integrated experiments (AMIE) provides objects representing all those entities, it has a `Function` object, which is completely generic. It has a collection of differential operators, such as `Gradient`, the usual ∇, `Differential`, $\frac{d}{dx}$, etc. It has a quadrature defined for the operators applied to the functions. It also provides assemblers and solvers to treat the system of equation.

Formally, a *function* is a map from $\mathfrak{R}^n \mapsto \mathfrak{R}$. It therefore takes n arguments, the values of x, y, z, etc. and uses them to compute a new value, which it return. This is the conventional definition of functions also in the field of computer science. However, the mathematical functions we wish to emulate have an additional property: they are differentiable. A further design constraint is that although the bases we consider are finite, there is an uncountable infinity of possible bases, thus it is not possible to have a library of predefined functions if we want to keep the generality of the code[1].

The `Function` class has been designed to be used in conjunction with another class, a virtual machine. The virtual machine is not explicit in the Galerkin description, but the Galerkin description implies that the function can be evaluated. The task of evaluating the function is devolved here to the virtual machine.

The function can be considered as a series of instruction to be applied sequentially to the arguments, with a memory of previous states, called a stack. The atomic operations to be applied are abstracted as `Token`s which each define an operation on the stack. For example, the add operation consists in taking the two last values on the stack and replacing them with a single new value, the sum of the two original.

All the operations can be uniquely identified by their name, so it becomes possible to form a function only from a series of words or symbols, read from a string.

```
Function f("x 1 -") ; // f(x) = 1-x
Function g("x sin") ; // g(x) = sin(x)
Function p = f*g ;    // p(x) = (1-x)*sin(x)
```

Listing 2: Functions from a string.

The functions defined are also part of a *space*, so operators on the functions must be defined, so the function object has the same mathematical properties as the analytical functions used in the Galerkin method.

The expression of functions as a series of atomic operations on a stack makes it very easy to further implement operators between the functions: for example, the operator $\times$ is the result of a single stack operation taking the two last values on the stack. The set of instructions from a first function will yield a single value at the bottom of the stack. If the instructions from a second function are appended, a second value is added to the stack. Further appending the $\times$ stack operation yields a function which is the product of the two original functions. The same principle is valid for any operator which can be used to combine functions.

[1]This is not just an exercise of style, allowing arbitrary bases to be defined in the framework is a prerequisite for a clean implementation of the XFEM.

Performance of such function object is easy to evaluate: the time taken to evaluate a function from the series of operations which define it, is essentially equal to the function call overhead multiplied by the number of atomic instruction,. Thus, it is desirable to optimise the number of tokens used in a function which will be called a lot. This is done using a simple compiler which identified repeated sets of instructions and replaces them by a single set and memory call-backs. The compiler also knows to simplify such operations as "add 0". a further optimisation consists in replacing series of tokens frequently occurring by single tokens. For example "multiply x by a value and add another" is a common occurrence in the basis functions. It is therefore efficient to perform this operation with a single token.

A Virtual Machine is responsible for taking a function, defined from a series of instruction and returning the value of the function from the argument. The core of the virtual machine here simply reads the tokens from the function in order, and uses them to compute the resulting value.

B.2 Integro-differential Operators

The virtual machine defines the integro-differential operations as special modes of evaluation of `Functions`. for example, given a quadrature and a function, the integral of a function can be computed (List. 3).

```
double VirtualMachine::ieval(const Function &f,
                             const GaussPointArray &gp)
{
   double ret = 0 ;
   for(size_t i = 0 ; i < gp.gaussPoints.size() ; i++)
      ret += eval(f, gp.gaussPoints[i].first)
             * gp.gaussPoints[i].second ;

   return ret ;
}
```

Listing 3: Numerical integral of a function from a quadrature scheme.

In a similar way, given a finite difference scheme, the derivative of a function can be similarly defined as a mode of evaluation of the virtual machine. More generally, all differential operators can be interpreted as a special kind of evaluation of a function, and therefore, additional operators can be added to the virtual machine as required. It is not possible to define an abstract syntax of the operators as for the functions, because the operators are typically continuous transformations of spaces, and the computer is a fundamentally discrete machine.

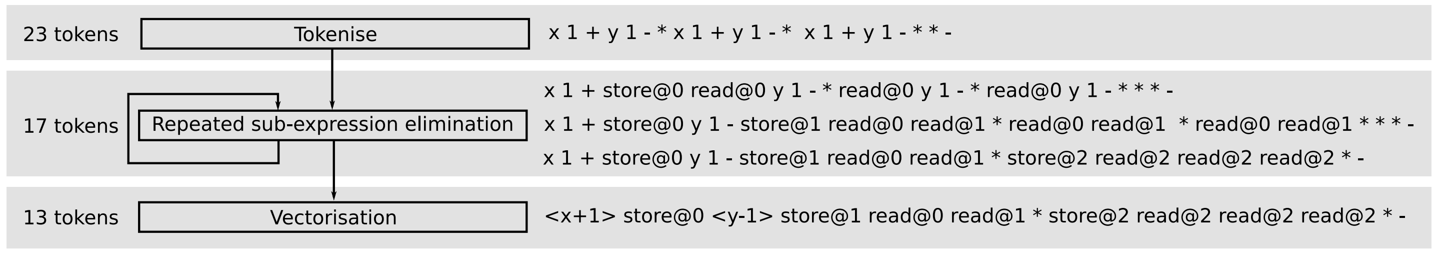
The finite difference schemes for most common differential operators are already implemented in AMIE. Notably, ∇, $\nabla\cdot$, $\nabla^T\mathbb{A}\nabla$ and simple differentials are implemented, which covers most of the usual linear partial differential equations of the first and second order. The derivation is done using a centred difference scheme. This scheme was chosen because it is stabler than high-order schemes and offers quadratic convergence.

The integration is done using quadrature schemes. However, exact quadrature schemes do not exist for all functions and some numerical error can occur. The quadrature is discussed in Sec. C.3.2 on page 193.

B.3 Compilation

The overhead due to the code being executed in a virtual machine is linear to the number of tokens. The compiler reduces the number of tokens in two passes. The first pass is tokenisation, which is the transformation of an expression in a set of atomic stack operations. The tokens are arranged in such a way that they can be read and the corresponding operations executed in sequence. This method is called static single assignment (SSA) [114, 115]. In the second pass, the compiler recursively searches for sub-expressions repeated more than twice. Each repeated sub-expression is transformed into a single sub-function, and the result is read instead of recomputed at each point where this expression previously occurred. This method is called common sub-expression elimination (CSE) [116]. In a final step, common series of operations, such as $x + 1$ are *vectorised*: they are transformed into single tokens which perform all operations at once (Fig. B.1).

Benchmarks run on code compiled this way show that if functions are reduced to a single token, the overhead is negligible compared to C++ compiled code. In general, the virtual machine is not a performance bottleneck, if the code it executes has been compiled and optimised to remove redundancies.



```
x 1 + y 1 - * x 1 + y 1 - *  x 1 + y 1 - * * -
x 1 + store@0 read@0 y 1 - * read@0 y 1 - * read@0 y 1 - * * * -
x 1 + store@0 y 1 - store@1 read@0 read@1 * read@0 read@1  * read@0 read@1 * * * -
x 1 + store@0 y 1 - store@1 read@0 read@1 * store@2 read@2 read@2 read@2 * -
<x+1> store@0 <y-1> store@1 read@0 read@1 * store@2 read@2 read@2 read@2 * -
```

Figure B.1: The overhead due to the code being executed in a virtual machine is linear to the number of token. The compiler reduces the number of tokens in two passes. After tokenisation, which is the transformation of an expression in a set of atomic stack operations, the compiler recursively searches for sub-expressions repeated more than twice. Each repeated sub-expression is transformed into a single sub-function, and the result is read instead of recomputed at each point where this expression previously occurred. In a final step, common series of operations, such as $x + 1$ are *vectorised*: they are transformed into single tokens which perform all operations at once.

APPENDIX C

Implementation and Tests of Various Enrichments

In this appendix are presented some commonly used enrichment schemes which were implemented using Automated meshing for integrated experiments (AMIE). In a second part, an adaptive quadrature generation method is presented.

C.1 LEFM Cracks

Linear elastic fracture mechanics cracks are the reason why the extended finite element modeling (XFEM) was first developed. Meshes, and mesh dependence makes it difficult to find good crack paths. Further, cracks induce sharp discontinuities in the solution space. Along their path, but also at the tip, where linear elastic fracture mechanics predicts a singular stress field. They pose difficult numerical problems, in terms of their geometrical representation, to obtain satisfactory quadratures at the crack tips, and the fact that near the tip, two different enrichments must be combined.

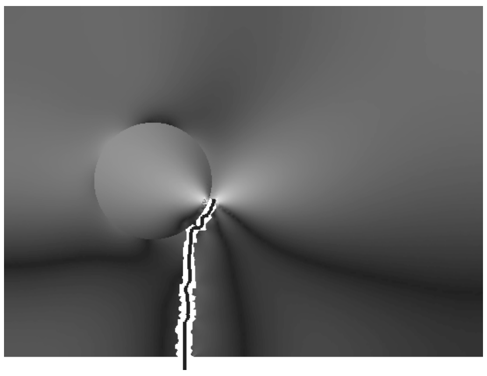


Figure C.1: Propagating crack in a simple setup with a single hard inclusion. The inclusion to matrix Young modulus ratio is 4.

The enrichment functions used for the discontinuity along the crack path are simply Heaviside distributions. The linear interpolation functions are N_i, and the Heaviside function is defined as:

$$H(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \text{ is “above” the crack} \\ 1 & \text{otherwise} \end{cases} \tag{C.1}$$

The enrichment at a node $\mathbf{p}$ is therefore:

$$h_i(\mathbf{x}) = (H(\mathbf{x}) - H(\mathbf{p}))N_i \tag{C.2}$$

At the crack tip, four different functions need to be used: $\{\sqrt{r}\sin(\frac{\theta}{2}), \sqrt{r}\cos(\frac{\theta}{2}), \sqrt{r}\sin(\frac{\theta}{2})\sin(\theta), \sqrt{r}\cos(\frac{\theta}{2})\sin(\theta)\}$. Those are the first four elements for the Fourrier development of the linear elastic fracture mechanics solution for the displacement field at the crack tip. The enrichment functions are therefore

$$\begin{aligned}
h_i^0(\mathbf{x}) &= \left(\sqrt{r(\mathbf{x})}\sin\left(\tfrac{\theta(\mathbf{x})}{2}\right) - \sqrt{r(\mathbf{p})}\sin\left(\tfrac{\theta(\mathbf{p})}{2}\right)\right) N_i \\
h_i^1(\mathbf{x}) &= \left(\sqrt{r(\mathbf{x})}\cos\left(\tfrac{\theta(\mathbf{x})}{2}\right) - \sqrt{r(\mathbf{p})}\cos\left(\tfrac{\theta(\mathbf{p})}{2}\right)\right) N_i \\
h_i^2(\mathbf{x}) &= \left(\sqrt{r(\mathbf{x})}\sin\left(\tfrac{\theta(\mathbf{x})}{2}\right)\sin\left(\theta(\mathbf{x})\right) - \sqrt{r(\mathbf{p})}\sin\left(\tfrac{\theta(\mathbf{p})}{2}\right)\sin\left(\theta(\mathbf{p})\right)\right) N_i \\
h_i^3(\mathbf{x}) &= \left(\sqrt{r(\mathbf{x})}\cos\left(\tfrac{\theta(\mathbf{x})}{2}\right)\sin\left(\theta(\mathbf{x})\right) - \sqrt{r(\mathbf{p})}\cos\left(\tfrac{\theta(\mathbf{p})}{2}\right)\sin\left(\theta(\mathbf{p})\right)\right) N_i
\end{aligned}$$

Although XFEM cracks are very good models for single cracks, some caveats must be noted. In cases were the cracking density is important, crack paths will intersect and cracks will merge. *Ad hoc* heuristics can be implemented to that effect, but it is difficult to have an energetically and numerically satisfying model for merging cracks. The enrichment functions have to be applied to a zone around the crack tip, and it is not clear what the code should do in the event that this zone crosses another crack.

The shape functions used for the crack tips cause the finite element matrices to have very bad condition numbers, and therefore significantly affect the convergence of problems where cracks are present. The initiation of cracks is difficult, and a transition from a damage state to an enriched state should be defined. In the case of problems where the cracking is dense, mesh independence is hard to ensure: crack density is capped by the element fineness in practise.

A further difficulty comes from the simultaneous growth of multiple cracks. A criterion for growth must be determined which takes into account the various cracks simultaneously growing. A heuristic which is frequently used assumes that crack growth is stable and that the total crack increment is fixed. The growth of each individual crack is tested by computing the J-integral around the crack tip, and the crack growth increment is divided equally amongst the cracks. A benchmark setup was used to verify the accuracy of our implementation (Fig. C.2).

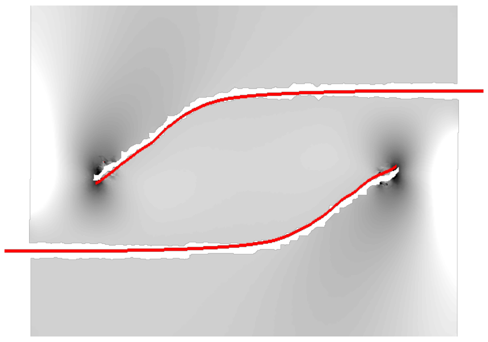


Figure C.2: Propagating cracks. This numerical experiment highlights the interaction between cracks, as well as concurrent crack propagation.

C.1.1 Sliding Interfaces

Sliding interfaces are discontinuities in the displacement field only in the direction of the interface. With a friction coefficient, they are a good model for a closed crack.

The transition from sliding interface to opening crack is however discontinuous. This makes this problem very non-linear and also a good benchmark for non-linear solver stability. We have implemented a closing crack using the penalty method: if an element is in compression along the normal of the crack, this induces a virtual force which prevents the penetration of a part of the element in the other.

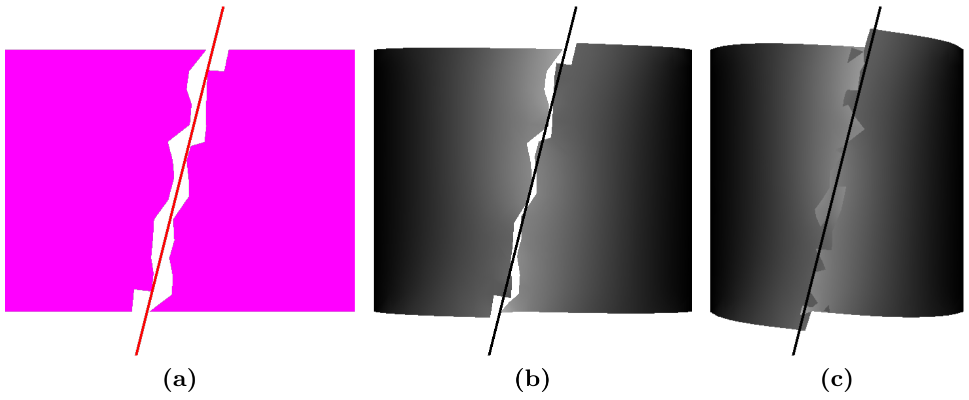


Figure C.3: Sliding interface simulation. The enrichment is a hard discontinuity projected normally to the crack path. Closing behaviour is hard to reproduce because it is highly non-linear.

The implementation showed that this enrichment could successfully model closing cracks and validated the non-linear solver[1]. Although for the simulation of alkali-silica reaction (ASR) XFEM was finally not used to model the damage, this implementation highlights the flexibility of the framework, and provides a validation for the non-linear solver.

C.2 Conclusion

The implementation and test of various enrichments has also been presented in this appendix as examples of the possibilities offered by the framework. For the simulation of ASR, only the soft discontinuity was used. The other enrichment types (cracks, sliding interfaces) were used as benchmarks as they are more demanding in terms of robustness.

C.3 Conforming Quadrature Generation

A crucial aspect of the XFEM is the quadrature. Because the basis function are not polynomials, there is no *a priori* quadrature such as the

[1]This behaviour is higly non-linear and discontinuous for small displacements. This makes the problem badly conditioned and convergence can only be obtained with a robust solver.

Gauß or Gauß-Lobato quadratures which can be used which will yield an exact solution. The introduction of singular shape functions further affects the convergence of the quadrature.

Although methods have been proposed with excellent convergence at the tip, they are limited to certain element types. Introducing such quadrature schemes as a special case for those elements types is therefore not a satisfyingly generic solution. A more general solution consists in the implementation of an automatic quadrature generator with refinement.

In many problems, features can overlap. For example, a crack can cross the boundary of an inclusion. In such cases, any number of enrichments might overlap in a single element. The quadrature must take all geometries into account, because Gauß points cannot lie on the boundaries of enrichments, which are typically the loci of discontinuities.

In Amie, the geometries of the enrichment features are sampled as for the main mesh, and the discretisation points are used to generate a submesh in the element where the quadrature is being computed. This approach ensures that the quadrature is conforming to the geometry considered and is general: any geometrical primitive defined in the geometry helper library can be used in this way.

This approach produces a conforming mesh of the geometry described by the enrichments.

C.3.1 Adaptive Refinement Algorithm

Although a conforming quadrature is generally produced, it might not be of a sufficiently high order to capture the shape functions. Thus an adaptive strategy is implemented which increases the quadrature order, using a so-called *h-p* approach.

In the first step, the conforming geometry is refined using a quad-tree-approach: the elements are subdivided at each iteration, yielding a mesh of characteristic size one half of that at the preceding step. Once the convergence is stable, which is defined as two successive steps have lowering the relative error, the final convergence is obtained by varying the quadrature order of each of the sub-elements.

The criterion used to check the convergence is the convergence of the integral of the enrichment function which has the highest gradient. A more exact method would involve computing the elementary matrix at each step of the refinement and checking the convergence. However, this approach would be prohibitively expensive.

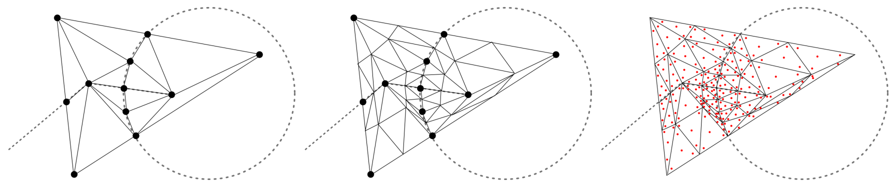


Figure C.4: Steps in the generation of a quadrature. A conforming mesh is generated as a function of the geometrical base of the enrichments. This mesh is refined using an adaptive quad-tree algorithm. The Gauß points are kept, and the sub-mesh discarded.

Although XFEM is a very flexible method, which allows partial mesh independence, it is not without drawbacks. The convergence rate of the solver is affected by the greater condition number induced by the elementary matrices of enriched elements.

This effect on the condition number is due to the fact that the extended basis is not orthogonal. The expected convergence is therefore $\mathcal{O}(n^{frac12})$ instead of $\mathcal{O}(n)$ which is the expected convergence rate for the conjugate gradient algorithm. Fig. C.5 illustrates this effect when comparing two identical setups, one with a meshed inclusion and one with an enriched inclusion.

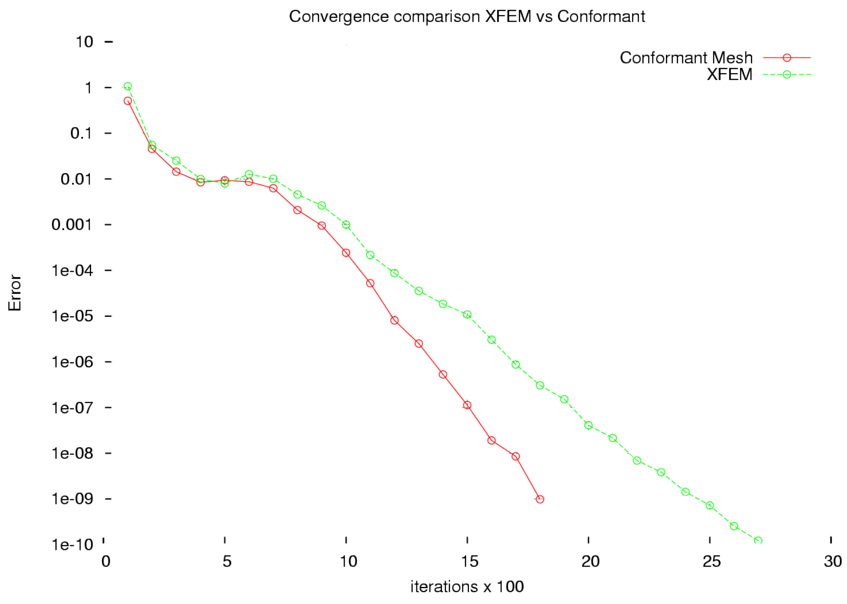


Figure C.5: Difference in convergence rates between enriched and conforming cases. The convergence rate of the enriched case is half that of the conforming.

Table C.1: Number of Gauß points used in a Monte-Carlo integration scheme versus the number of iterations to convergence.

# of GP	Infinity norm	Iterations	Convergence
16	$4.491\,68 \times 10^7$	44 108	No
32	$3.356\,50 \times 10^7$	44 108	No
34	$4.030\,39 \times 10^7$	39 164	Yes
36	$4.191\,39 \times 10^7$	29 856	Yes
40	$4.126\,30 \times 10^7$	31 675	Yes
48	$3.534\,43 \times 10^7$	28 584	Yes
56	$3.864\,30 \times 10^7$	24 400	Yes
64	$3.816\,27 \times 10^7$	18 369	Yes
92	$4.021\,56 \times 10^7$	14 025	Yes
128	$4.012\,38 \times 10^7$	14 833	Yes
256	$3.752\,90 \times 10^7$	13 273	Yes
512	$3.844\,02 \times 10^7$	8891	Yes
1024	$3.863\,58 \times 10^7$	8600	Yes

The matrix condition number κ is the predominant factor in determining the time needed for a conjugate gradient iteration. The bounds of the error as a function of the condition number are [117]:

$$||e_i||_A \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^i ||e_0|| \tag{C.3}$$

To test the link between the condition and the error of the quadrature, we have created a setup where a crack lies very near the boundary of an element, such that the matrix would be ill-conditioned even if the integration be exact.

Further, as no exact quadrature scheme exists in general for the enrichment used in the study[2], a criterion had to be determined to quantify the error introduced. This criterion was used as a test to decide if the refinement of the quadrature in the enriched elements was sufficient. A readily available number is the number of iterations to convergence. A conjugate gradient descent is expected to converge to a specified error in a number of iterations proportional to the condition number of the matrix.

The Monte-Carlo integration scheme[3] has an expected convergence of $\mathcal{O}(n^{\frac{1}{2}})$. We assumed this relation to be verified for the amount of points

[2]There exists no general quadrature for mixed power/harmonic functions.

[3]In the Monte-Carlo quadrature, points are randomly placed and equally weighted.

used in this experiment and could plot the number of iterations as a function of the relative quadrature error (Fig. C.6). This figure shows that if the Gauß points are not correlated to the position of the enrichment-generating feature, a very large number is required. Further, it shows that the number of iterations to convergence can be used as a measure of the error as they are strongly correlated.

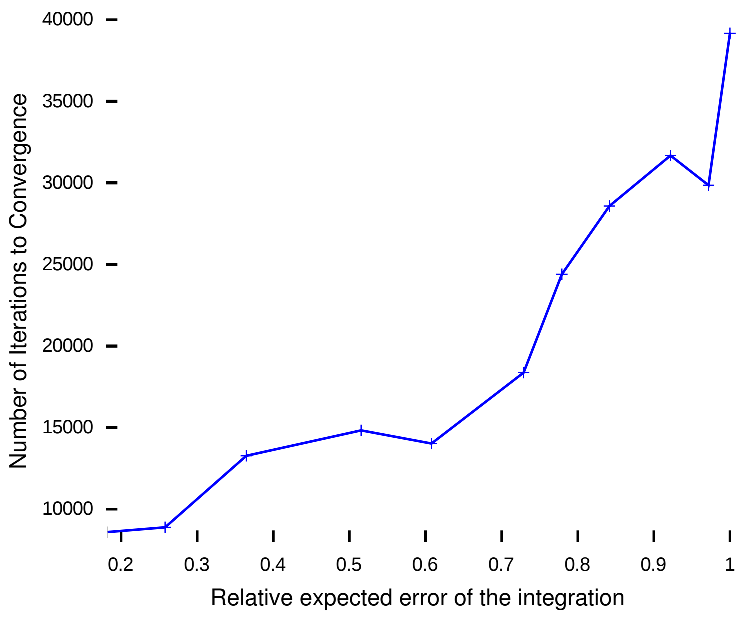


Figure C.6: Number of iterations to convergence as a function of quadrature error.

The convergence of the number of iterations required in a setup could then be used as a a proof that the integration step was sufficiently precise. Using this criterion, it was possible to optimise the refinement algorithm to balance the cost, which depends linearly on the number of Gauß points used and the precision, which could be measured as the number of iterations required to converge.

C.3.2 Comparison with High-order Regular Quadrature

An alternative to the generation of conforming quadratures is to produce a fine regular grid of Gauß points and integrate on them. This method is easily implemented and is said to offer good results. However, it suffers from some drawbacks: if the source for the enrichment has a boundary very close to the element boundary, there might not be integration points on both sides.

Further, although this is unlikely, a Gauß point might lie very close to a singularity. For these reasons, in the case of numerous enrichments, automatically generated, such a quadrature might be unsafe to use, as it can lead to singular matrices. Indeed, although the error is proportional to the area of the sub-elements, it is also bounded by the largest error in any of them. In the case of singular enrichments, this error is unbounded.

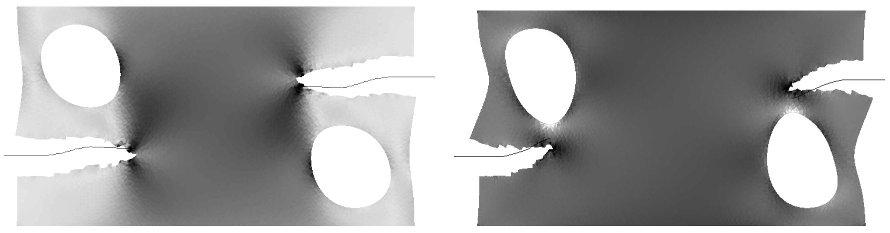

(a) Adaptive refinement **(b)** Homogenous quadrature

Figure C.7: Crack paths obtained with two quadrature methods: adaptive and homogeneous. The homogeneous quadrature is not conforming and yields higher strains than the adaptive. Therefore it tends to overestimate the curvature of crack paths.

To compare the two quadrature methods, a simple symmetric setup with two cracks and two pores under traction was set up. The crack paths computed from both methods were compared. It was observed that the regular grid quadrature yielded excessive crack curvature. This result was expected as there is an inherent bias in the sampling method which causes the crack to tend to pass element boundaries. This confirmed the need for an integrated approach which also provides geometry routines.

APPENDIX D

Mesher Benchmarks

In this appendix are shown some examples of the meshes which the integrated mesher of Automated meshing for integrated experiments (AMIE) can produce. These meshes have been produced with no user intervention other than the specification of the geometry, or rules for the generation of geometry. The tests were used to stress-test the mesher and verify its stability[1].

D.1 Concrete Microstructure

Mortar and concrete microstructure generation and meshing are central to this thesis. Some ultra-dense microstructures were generated and meshed. On Fig. D.1 the input microstructure and resulting mesh are displayed at various zoom levels. Such a microstructure contains more aggregates than present in typical mix designs.

[1]Due to the multi-scale nature of the geometries considered, the mesher is susceptible to numerical rounding errors.

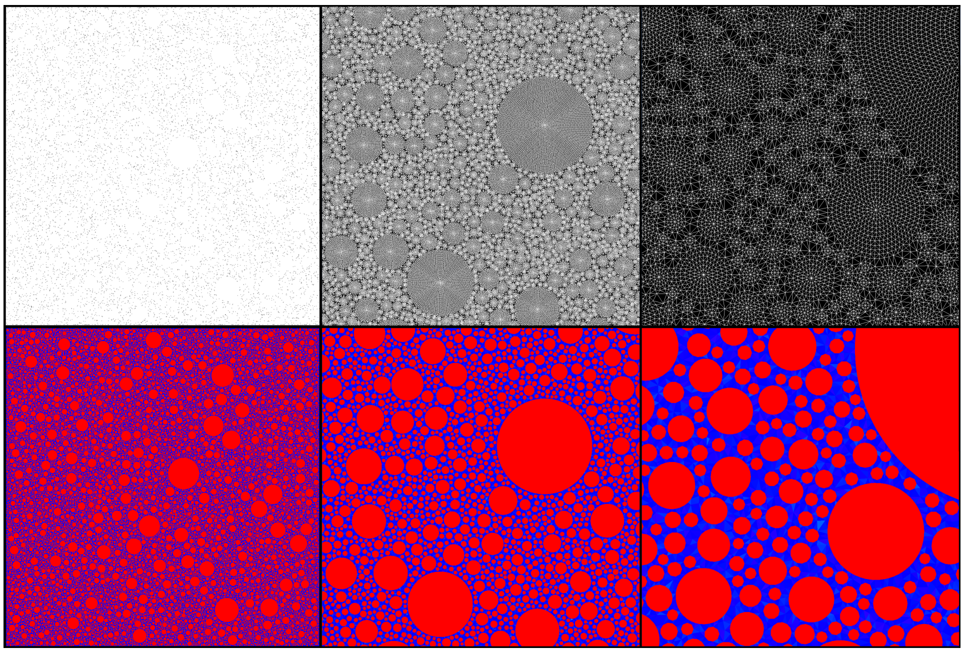


Figure D.1: Mesh of a mortar-like microstructure. The input geometry is shown in greys at various levels of zoom, in black and white, the corresponding mesh. The very dense mesh obtained here shows the mesher is robust with respect to numerical instability.

In three dimensions the limiting factor becomes the available memory, and it is not possible to fully represent the microstructure of mortar samples. However very close approximations can be obtained with AMIE's mesher.

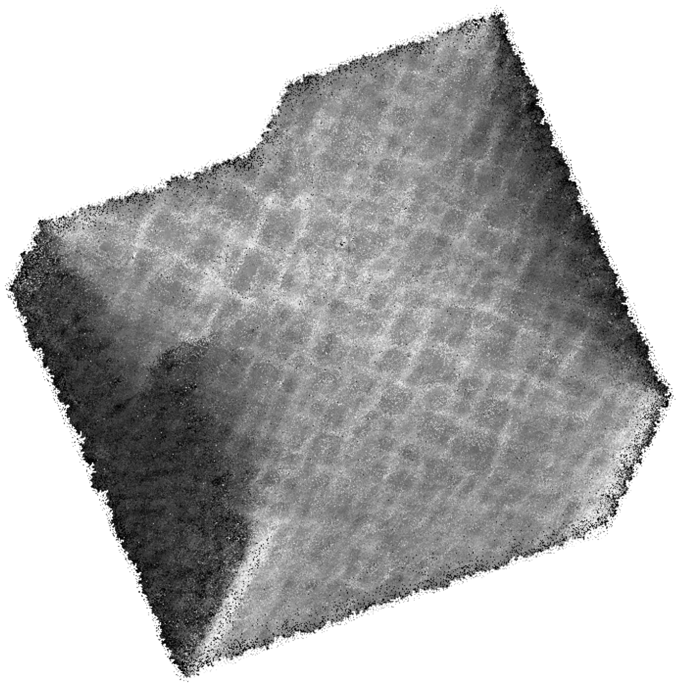


Figure D.2: The element density of a 3D mesh is apparent in the figure: each point is an element, there are approximately six million elements in this mesh. The color represents stress intensity. This microstructure is filled with 96000 inclusions, for a filling ratio of 57%.

The real filling in terms of volume of the aggregates is approximately 65% for mortars. The mesher could mesh microstructures with a filling factor of up to 57%. The limitation was not the mesher but the available memory.

D.2 Bone-Like Structure

Highly porous structures are delicate to mesh, because small errors in the representation of the geometry may yield dramatic effects because the solid skeleton of the sample might not be connected anymore.

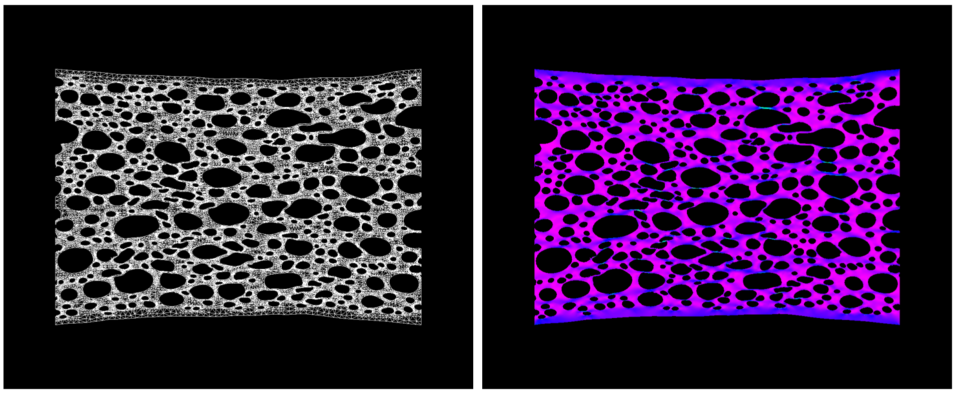


Figure D.3: Mesh of a bone-like microstructure. The high proportion of holes in this mesh makes it very sensitive to errors in the geometrical conformance of the mesh. The display of the strain values shows that there is no error in the mesh: no dangling is apparent.

Despite many pores located close to each other the mesher did not produce significant errors on the meshing: all solid connections were correctly meshed. The pores are also meshed and the elements kept for the fast search routines. They are marked "void" however and are therefore not assembled.

D.3 Mic Microstructure Meshing

μic is a cement hydration modelling platform which outputs very complex microstructure composed of many overlapping spheres [118]. Due to the very important range between the largest and smallest sphere (almost four orders of magnitude) such geometries pose significant challenges to meshers. The Fig. D.4 shows that the integrated mesher from AMIE performs rather well and offers significant improvement over regular meshing approaches.

For this test, only the outer shells of solid material were used to form the negative of a pore network which was meshed. A diffusive behaviour was then attributed to the elements. Two setups were used. In the first, a regular tetrahedral mesh was produced and the elements which centre did not lie in a solid particle were kept. In the second setup, the spheres representing the particles were discretised explicitly in AMIE and meshed. The intersections of all spheres were computed explicitly which produced additional points to be entered in the mesh. All elements were then attributed transport properties.

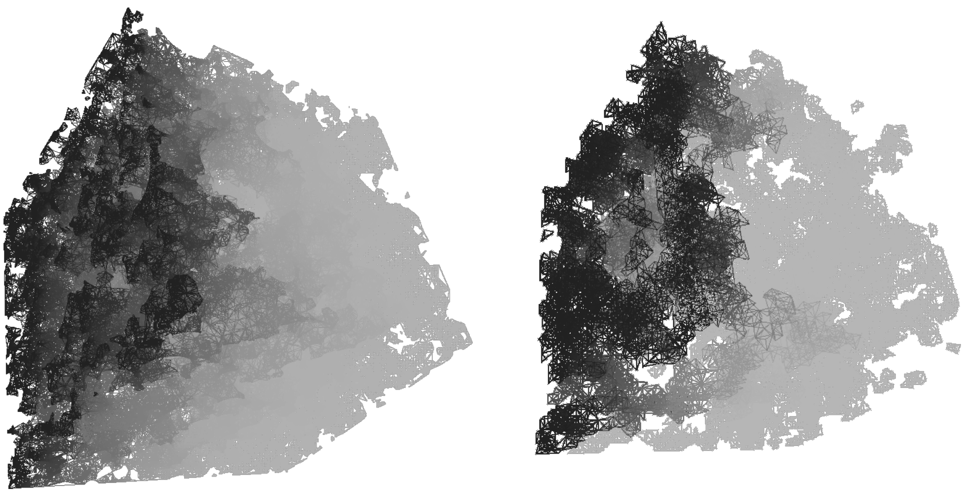


Figure D.4: Comparison of regular and conforming meshes. The conforming mesh (left) displays a much higher level of detail than the regular mesh (right). This has significant impact on simulations of diffusion through the pore network (transitory phase shown here).

This kind of geometry tends to stress the limits of meshers as the intermediate steps of the meshing involve the formation of near-degenerate elements. This problem was solved by imposing a minimum distance between nodes inserted in the mesh, effectively introducing a resolution limit, which is however much better than the resolution of the regular mesh. Another working method to avoid inserting overly close points is not to insert supplementary points at the intersections of the spheres.

APPENDIX E

Damage Benchmarks

In this appendix are presented some model problems which were used to test the damage model implementation.

E.1 Hard inclusion

Analytical models are limited when the phase contrast is too high. In such a setup, a crack in the soft phase will branch at a distance from the hard phase interface.

A setup with a single centred inclusion, 1000 times stiffer than the matrix was tested. The loading was pure vertical strain imposed on the boundaries (Fig. E.1).

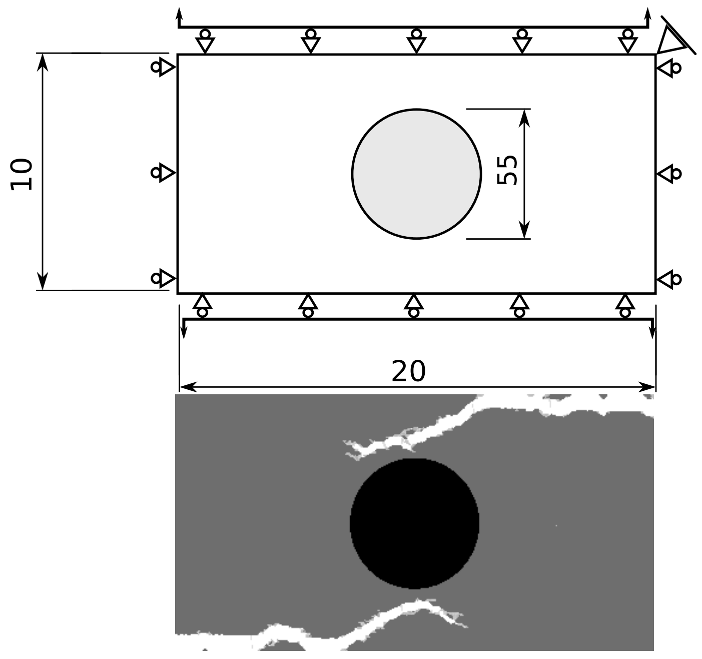


Figure E.1: Setup and crack pattern observed with the tension test with a hard inclusion and a phase contrast of 1000. The central (as opposed to the expected axial) symmetry observed is due to a slight numerical imprecision, ultimately caused by the boundary conditions.

On Fig. E.1 the cracks initiate at a distance from the inclusion and branch laterally. The crack paths are not axisymmetric. This comes from the the fact that to prevent rigid body motion, at least one node had to be fixed in both directions, making the problem numerically slightly asymmetrical. However the expected features are well present: the rounded v-shape initiating at a distance from the inclusion, the lateral branching of the cracks, and the deflection near the top and bottom boundaries (See [119, 120, 121] for various aspects of such a problem).

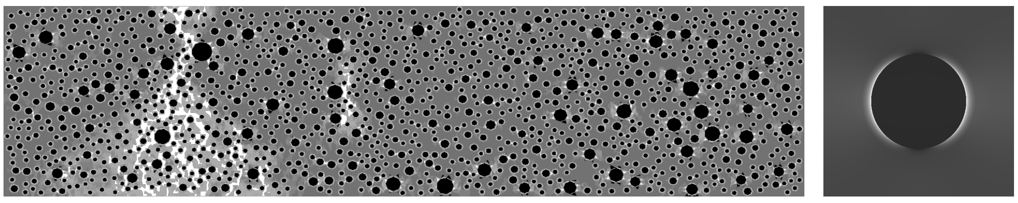
The branching is an essential property of damage models, as opposed to embedded discontinuity models, and this advantage is retained here: although the elements are ordered and damaged one at a time, branching still occurs.

E.2 Composite problems

E.2.1 Study of the influence of the material model of the ITZ

The interface transition zone (ITZ) between aggregates and paste in concrete is known to be the source of the softening observed in this material [122]. The ITZ is caused by the arrangement of cement grains around aggregates, which induces a gradient of porosity from the aggregate surface to the paste, the so-called "wall effect". In numerical simulations, this zone is usually modelled as a homogeneous region with different properties to the bulk. However, the gradient can also be explicitly modelled. In either case, the zone is very thin: approximately 20 μm [123, 124]. Numerical experiments on the effect of such fine details are usually difficult to conduct. The simulations presented here serve to highlight the damage model's ability to cope with fine details in composite microstructures, and the results are not expected to match reality otherwise than in qualitative terms. Simulations were run with quadratic Lagrangian elements.

A displacement-controlled pure-tension test was performed on 4×16 simulated mortar bars with two material models for interface transition zones: a single averaged material and a zone with a gradient of mechanical properties. The particle size distribution of the aggregates is not realistic: it is extremely reduced compared to real mortar bars. Also, the ITZs are larger than in reality.


(a) Smooth ITZ (b) Detail

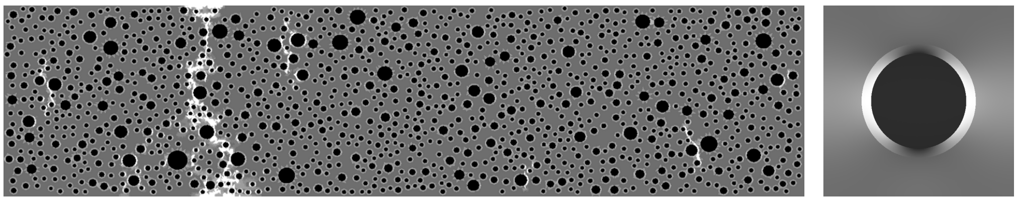
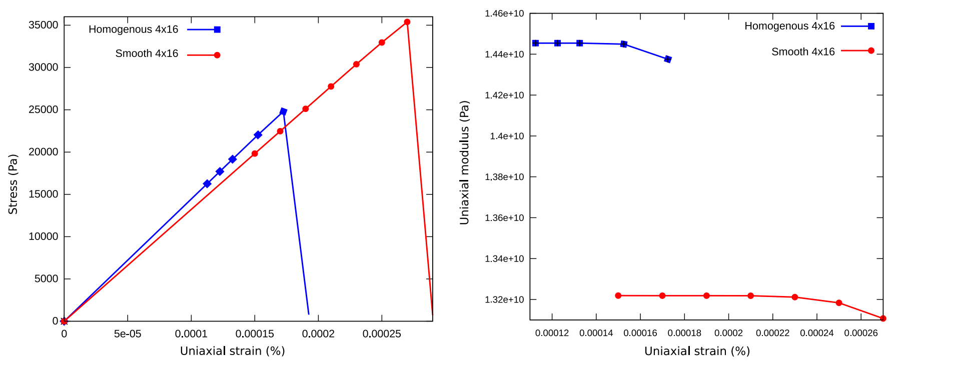

(c) Homogeneous ITZ (d) Detail

Figure E.2: State of the specimens after failure. The delamination in the case with the smooth ITZ is more marked (a). Despite the brittle behaviour of the individual elements, the damage does not localise in a single-element-thick band in either case but is rather spread out. Detailed views of the von Mises stress distribution in the smooth and averaged cases are shown. The fine details of the stress distribution are responsible for the localisation of failure. Such fine details would be lost when using smoothened fields.

The smooth gradient of properties in the ITZ led to a localised progressive delamination of the aggregates, which increased the compliance of the samples, and yielded an apparent macroscopic softening behaviour. Conversely, if the mechanical property of the ITZ is homogeneous, the delamination happens brutally at the onset of total failure. In the smooth model, the softening happens over three displacement steps, whereas in the homogeneous model, a single step of softening is observed.



(a) Stress-strain curve **(b)** Uniaxial stiffness

Figure E.3: Stress-strain curve of the specimen under tension (a), apparent modulus during test (b). These simulations were run to test the influence of the ITZ model on the failure behaviour of the specimen, but were not meant to be an accurate representation of real mortar samples, as only the largest aggregates are represented.

The critical stress for the sample with the smooth ITZ is higher than for the homogeneous model. This is because the critical stress for the sample depends on the maximum local stress level. In the smooth case, there are less points whith sharp phase contrast, and thus less potential points for the initiation of the sample failure.

The stress fields obtained are compared in Fig. E.2. As can be expected, the stress field varies more smoothly when the mechanical properties of the ITZ are not constant. But also, the average strain within the aggregate is smaller in the case of the smooth ITZ. This is because although the average mechanical properties are almost the same in both setups, the strains are more distributed in the smooth ITZ: the part of the ITZ which is immediately surrounding the aggregate can take a much larger amount of strain.

Fig. E.2 shows the different fracture paths of the two samples after failure. The main crack network is much more extensive in the smooth case, which also exhibits delamination of the aggregates. The delamination in Fig. E.2c is made more visible due to the fact that the whole damaged elements are displayed white, and should not be interpreted as the desolidarisation of all aggregates from the paste.

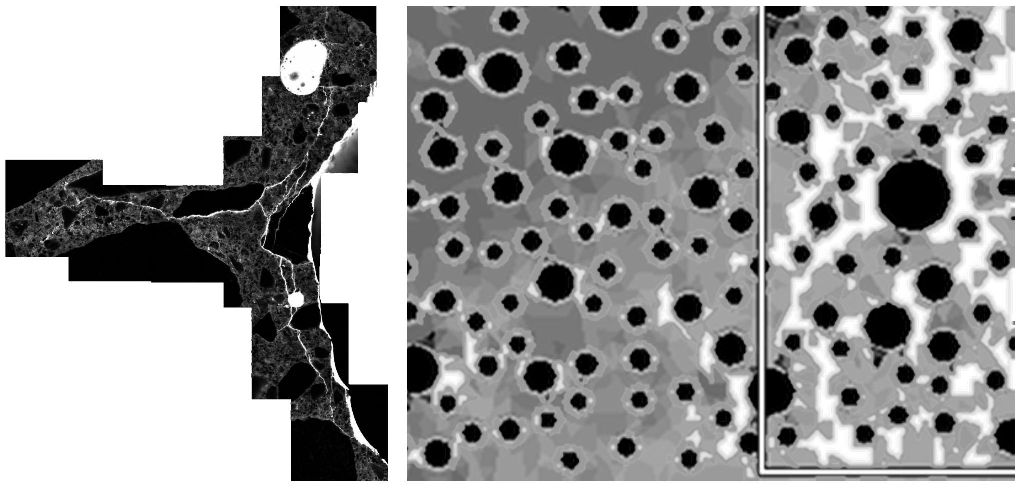

(a) Experiment (b) Simulation

Figure E.4: Crack pattern resulting from the failure in tension of a concrete sample on a single side of the main crack, from Baillion [125] (a). The experimentally observed degraded zone is approximately two centimetres thick, which is much closer to the results from the smooth model (one centimetre wide box shown for scale (b)). Various features of the experiment are well reproduced by the model: secondary cracks, delamination and the extent of the damaged zone. The ITZ in the model is larger than the real one, and is the source of more cracking.

The comparison of the width of the failure zone with results from the literature (Fig. E.4) shows good agreement in the observed extent of the damage around the main crack (around 2 cm on average) with the smooth model [125], despite the reduced particle size distribution. The presence of lateral secondary cracks running parallel to the main crack is also reproduced.

Damage computed from smoothed fields cannot capture the effect of the property gradient, especially if the radius for the smoothing is larger than the typical size of the ITZ. Thus an algorithm which can test for the effect of the model chosen for the ITZ is required to perform the numerical experiments presented above. Because the points of failure are numerous in a given sample, it is also possible to limit the radius of investigation of the algorithm.

The global component of the smoothed equivalent stress/strain fields used in usual global methods is quite large to ensure mesh independence. It is in any case larger than the typical ITZ size by at least two orders of magnitude, and thus those algorithms are unsuitable for such an investigation.

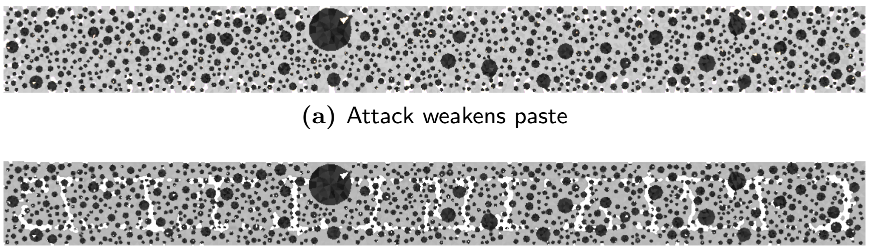
The model used for the ITZ uses the formalism of a functionally graded material. The computation of the fracture criterion using the original stress and strain fields can be performed at enough sampling points within the elements so that the effect of varying mechanical properties within an element can be correctly captured.

The propagation of XFEM cracks in such materials is still a research topic. Notably because the propagation criteria are typically based on Rice's J-integral, which is path dependent in those materials. However, the correct initiation of cracks would require the computation of a failure criterion at all potential locations for the initiation, and comparing among those locations which is the most likely candidate. The algorithm presented in this chapter could serve that purpose.

E.2.2 Degradation due to External Sulphate Attack-Like Processes

To highlight the ability of the described algorithm to cope with externally imposed changes in material properties, we did some simulations of external sulphate attack. External sulphate attack is a durability issue for concretes used in aggressive environment, such as the cover of sewage pipes. It is characterised by the progressive change of the phase makeup in the cement paste behind a reaction front, and swelling of the paste in that zone. These simulations are meant as demonstrations of the damage modelling in large setups, not as exact or physically relevant simulations.

From a mechanical point of view, this process can be modelled in two different ways: the degradation process can be assumed to weaken the paste, in that the critical load at which it fails becomes lower, or it can be assumed to change the mode of fracture of the paste, making it more ductile for example. Both mechanisms are described by Santanam and colleagues [126, 127].


(a) Attack weakens paste

(b) Attack changes failure mode of paste

Figure E.5: Failure mode of samples subject to external sulphate attack. The attack effect is studied under two hypotheses, that the attack weakens the cement paste, but does not change the failure mode (a), and that the attack changes the failure mode but does not affect strength (b).

Depending on the model chosen, the failure mode of samples will differ: if the criterion is weakened, the attacked zone of the material fails leaving the core intact, whereas if the external swelling layer becomes more ductile, the healthy core of the sample will fail (Fig. E.5). Both mechanisms have been observed in laboratory and field samples, indicating different processes might be occurring, depending on the kinetics of the reaction. Indeed, the mechanisms behind external sulphate attack are still the object of controversy.

Cyrille Dunant ing. dipl. epf

🖂 EPFL-STI-IMX-LMC cyrille.dunant@epfl.ch
Station 12 MXF210 – CH-1015 Lausanne – Switzerland
☎ +4121 624 17 33 or +4121 693 46 73

Principal skills: modelling, programming, FEM, image analysis, concrete technology.

Education

Ph.D. in Material Science – EPFL, 2009. Under the supervision of Prof. Karen L. Scrivener (*in progress*). Implementation of a XFEM framework, with companion solver, meshing and geometry library for the modelling of the alkali-silica–reaction (ASR) at the microstructure level. Development of an image analysis software for the statistical analysis of the damage state of concrete from back-scattered electron microscopy polished section images.

MSc. in Mechanical Engineering – EPFL, 2004. Diploma at the "Laboratoire d'Ingénierie Numérique" (LIN) under the supervision of Dr. A. Drotz. "Fractal extension of computing domains for boundary conditions at infinity."

Professional Experience

October 2005 – Present: Graduate student under Prof. K. L. Scrivener's supervision at EPFL's Laboratory of Construction Materials.
Research experience: Initiated and completed a project to model the micromechanical effects of ASR-induced degradation; established a novel computing framework for finite elements; helped develop and implement an *in-situ* XRD monitoring protocol for early hydration
Supervisory experience: Supervised a master's student and 2 undergraduate students' projects (4 months each).
Teaching experience: Taught laboratory courses for undergraduate students.
Outreach: Organised a workshop on the numerical tools developed during the thesis (the Amie XFEM framework), May 2009.
Outreach: Member of the organising comitee for CONMOD, June 2010.

April 2005 – October 2005: Intern at the Laboratory of Construction Materials.

Research experience: Rearchitectured and reimplemented the in-house FEM code from FORTRAN to C++.

July 2004 – February 2005: Intern at the Laboratory of Autonomous Systems 2 (ASL2).
Research experience: Implemented a real-time tracking software for robotics at the ASL2; built and configured a computer cluster (including custom load-balancing software).

January 2004 - June 2004: Diploma student under the supervision of Dr. A. Drotz at the LIN.
Research experience: Invented, implemented and tested a novel finite element type.

Other Experience

Student representative: For the Section and at the Faculty Council. Student representative at the Faculty IT Commission.

Extracurricular activities: Co-founder of EPFL's Linux user group.

Skills

Laboratory techniques: Electron microscopy, mechanical testing, ASR expansion tests.

Programming languages: C++, Perl, C, Java, Pascal.

Specific computing skills: FEM modelling, real-time and off-line image analysis, GUI with Qt™ and OpenGL™.

General computing skills: Linux, MacOS X, Windows. Office and LaTeX. Raster graphics with Gimp and Photoshop, vector graphics with Inkscape and publishing with Scribus.

Languages: Fluent (both written and oral) in French and English. Knowledge of German, written and spoken.

Publications

Peer-Reviewed Journal Papers

C. F. Dunant and K. L. Scrivener. Micro-Mechanical Modelling of Alkali-Silica–Reaction—Induced degradation unsing the amie framework. *Cement and Concrete Research*, 2009.

A. Wilson, E. Laurenti, G. Oser, R. C. van der Wath, W. Blanco-Bose, M. Jaworski, S. Offner, **C. F. Dunant**, L. Eshkind, E. Bockamp, P. Lio, H. R. MacDonald and A. Trumpp. Hematopoietic Stem Cells Reversibly Switch from Dormancy to Self-Renewal During Homeostasis and Repair. *Cell*, 135:1–12 December 2008.

C. F. Dunant, P. N. Vinh, M. Belgasmia, S. Bordas and A. Guidoum. Architecture Tradeoffs of Integrating a Mesh Generator to Partition of Unity Enriched Object-Oriented Finite Element Software. *Revue Européenne de Mécanique Numérique*, 16:237–258, March 2007.

S. Bordas, V. P. Nguyen, **C. F. Dunant**, H. Nguyen-Dang, and A. Guidoum. An Extended Finite Element Library. *International Journal for Numerical Methods in Engineering*, 71:703–732, January 2007.

Manuscripts Submitted

C. F. Dunant, S. Bordas and K. L. Scrivener. An Algorithm for Computing the Locus of Irreversibility in Heterogeneous Materials. *Journal of the Mechanics and Physics of Solids*

Peer-Reviewed Conference Papers With Talks

A. Quennoz, **C. F. Dunant** and K. L. Scrivener. Deconvolution Method for Heat Evolution Curves of Model Cements. CONMOD *2010,* Lausanne (Switzerland), 2010.

C. F. Dunant, K. L. Scrivener and S. P. A. Bordas. A simple Algorithm to Simulate Energy Minimal Fracture and Damage: Applications to Complex Microstructures. XFEM *2009,* Aachen (Germany), 2009.

A. Quennoz, E. Gallucci, **C. F. Dunant** and K. L. Scrivener. Influence of the gypsum amount on the hydration of tricalcium aluminate in C_3A-gypsum and in alite-C_3A-gypsum systems. IBAUSIL *2009,* Weimar (Germany), 2009.

C. F. Dunant, A. Guidoum and K. L. Scrivener. Modélisation Microstructurelle de la RAG. RF^{2}B, Lausanne (Switzerland), 2008.

C. F. Dunant, A. Guidoum and K. L. Scrivener. Micro-Mechanical Modelling of ASR. 13th ICAAR, Trondheim (Norway), 2008.

C. F. Dunant and K. L. Scrivener. A New Modelling Framework for Meso-Scale Simulation of Concrete. CONMOD *08,* Delft (The Netherlands), 2008.

Invited Talks

C. F. Dunant and K. L. Scrivener. Implementation of a Fully-Generic Galerkin Simulation framework. WWCM08/ECCOMAS *2008,* Venice (Italy), 2008.