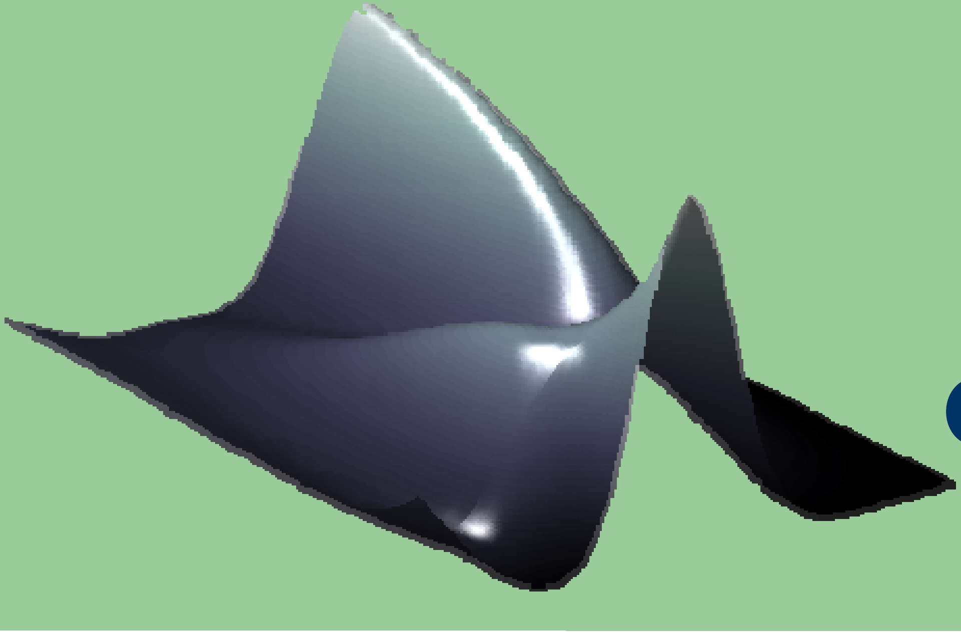
Soft- & Hard Modelling of Kinetic Data



An introduction to the basic principles for the analysis of calorimetric & spectroscopic data

- Julien Billeter & Bobby Neuhold

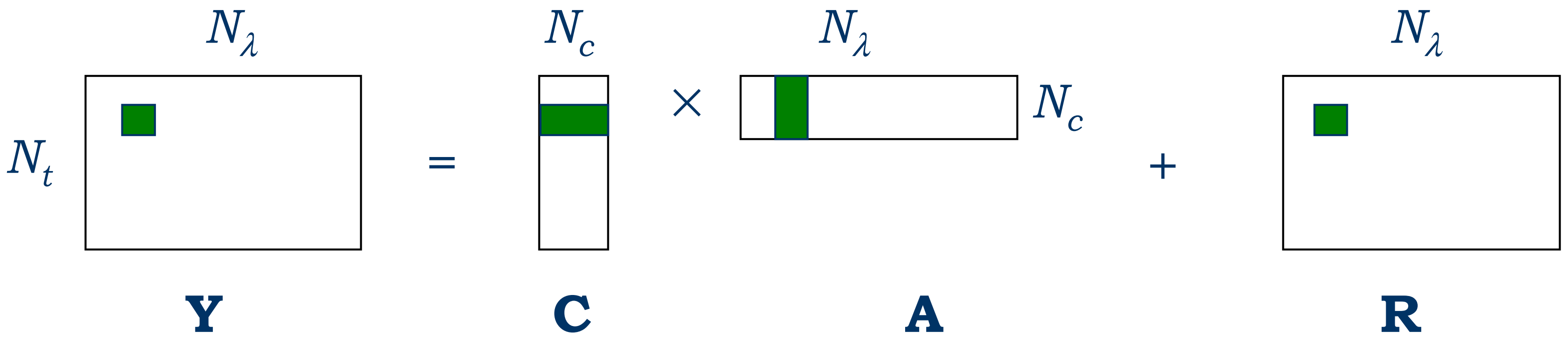
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Safety and
Environmental
Technology Group

Kinetic Data Structures

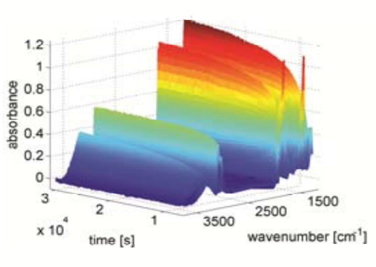
Kinetic Data Structures

- **Spectroscopy**: Beer’s law in elegant matrix notation, absorbance (**Y**) is proportional to the concentrations (**C**)

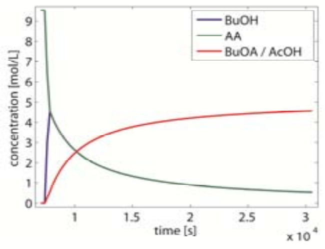


Kinetic Data Structures

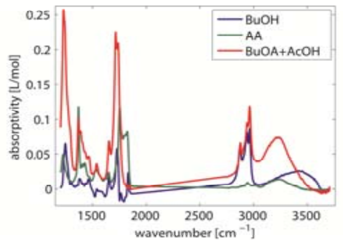
- **Spectroscopy**: Beer’s law in elegant matrix notation, absorbance (**Y**) is proportional to the concentrations (**C**)



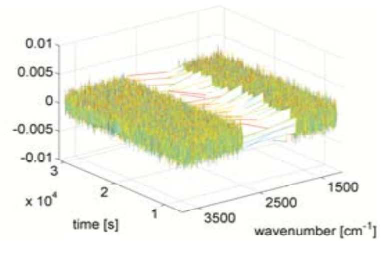
Y



C



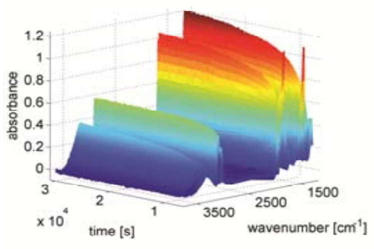
A



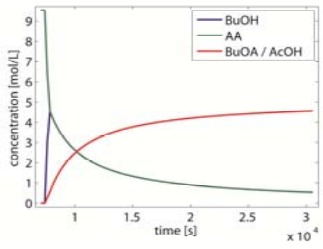
R

Kinetic Data Structures

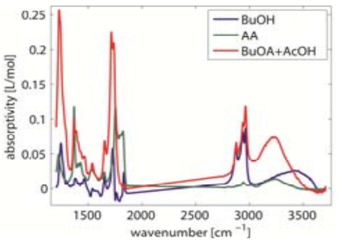
- **Spectroscopy**: Beer's law in elegant matrix notation, absorbance (**Y**) is proportional to the concentrations (**C**)



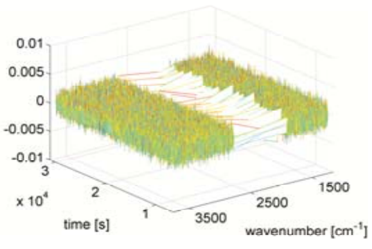
Y



C

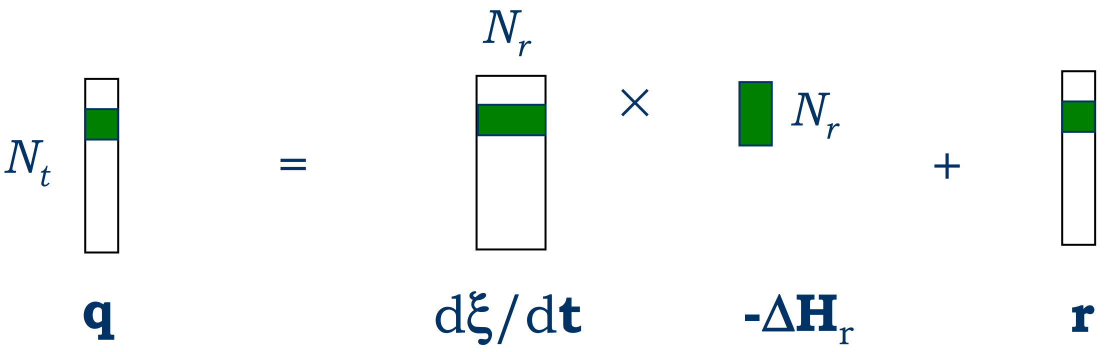


A



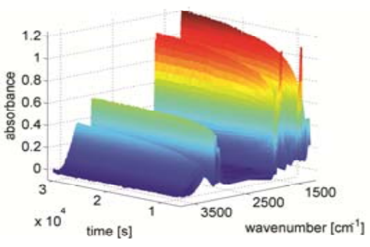
R

- **Calorimetry**: heat release (**q**) is proportional to the change in the reaction extent (dξ/d**t**)

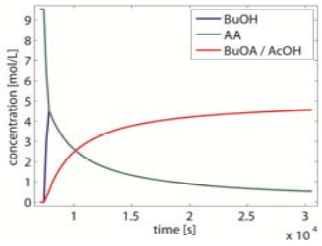


Kinetic Data Structures

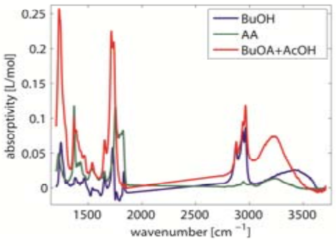
- **Spectroscopy**: Beer's law in elegant matrix notation, absorbance (**Y**) is proportional to the concentrations (**C**)



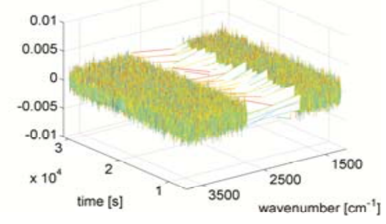
Y



C

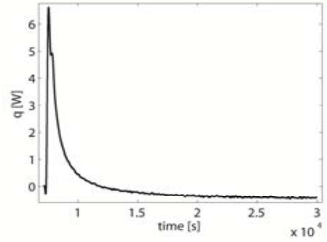


A

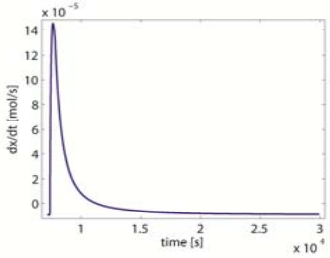


R

- **Calorimetry**: heat release (**q**) is proportional to the change in the reaction extent (dξ/d**t**)



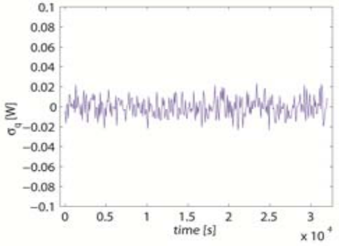
q



dξ/d**t**

43 kJ/mol

$-\Delta \mathbf{H}_r$



r

Soft- & Hard Modelling of Kinetic Data

Soft- & Hard Modelling of Kinetic Data

Soft Modelling (Part 1)

- Modelling experimental kinetic data based only on simple ‘a priory’ knowledge on the data structure and the results
  - Multivariate (multi wavelength spectroscopic) data and linear dependence of concentrations and data signal (Beer’s law)
    - non-negativity of concentrations and species spectra
    - closure, unimodality, etc
- cannot be applied to calorimetry data (power signal is univariate)

Soft- & Hard Modelling of Kinetic Data

Soft Modelling (Part 1)

- Modelling experimental kinetic data based only on simple ‘a priory’ knowledge on the data structure and the results
  - Multivariate (multi wavelength spectroscopic) data and linear dependence of concentrations and data signal (Beer’s law)
    - non-negativity of concentrations and species spectra
    - closure, unimodality, etc
- cannot be applied to calorimetry data (power signal is univariate)

Hard Modelling (Part 2)

- Modelling experimental kinetic data based on a parameterised physical-chemical ‘hard’ model
- rate law defines the concentration profiles of the contributing species as a function of the rate constants
- applicable to both, calorimetry & spectroscopy (univariate & multivariate)

Soft- & Hard Modelling of Kinetic Data

Part 1: Soft Modelling

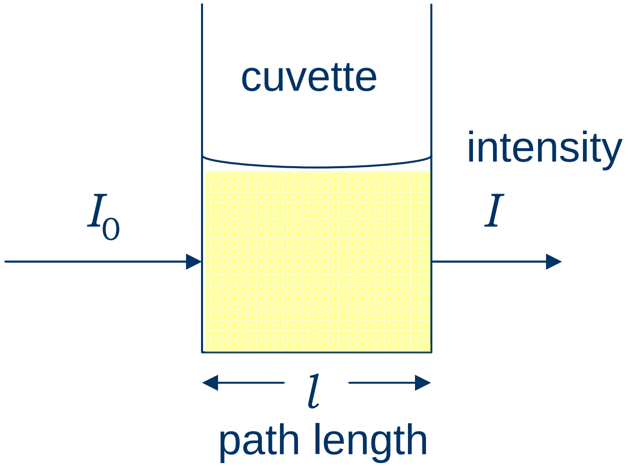
Topics

- Absorption spectroscopy
  - Beer’s law in elegant matrix notation ($\mathbf{Y} = \mathbf{C} \times \mathbf{A}$)
  - Non-unique factorisation of **Y** / rotational ambiguity
- Principal Component Analysis (**PCA**)
  - Abstract Factor analysis (**AFA**) by Singular Value Decomposition (**SVD**)
  - Chemical rank of the measurement matrix
  - The number of absorbing species
- Evolving Factor Analysis (**EFA**)
  - Evolutionary rank analysis by repeated SVD of sub matrices of **Y**
  - The ‘Appearance’ & ‘Disappearance’ of absorbing species
- Multivariate Curve Resolution by Alternating Least-Squares (**MCR-ALS**)
  - Model-free iterative decomposition of $\mathbf{Y} = \mathbf{C} \times \mathbf{A} + \mathbf{R}$
  - Ideas, principles, limitations

Absorption Spectroscopy – Beer's Law

Absorbance *y* at wavelength λ:

$$y_\lambda = -\log\left({}^{I}\!/\!_{I_0} \right)_\lambda$$



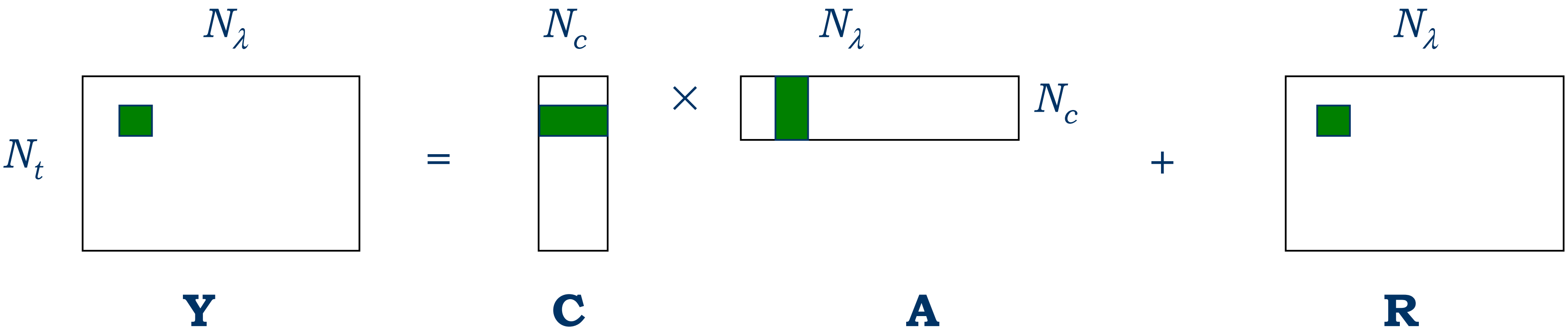
- Absorbance signal y_λ is linearly dependent on contributing species concentrations c_k, the corresponding coefficients are the molar absorptivities $a_{k,\lambda}$ that form the pure species spectra

Beer's Law:

$$y_\lambda = \sum_{k=1}^{N_c} c_k a_{k,\lambda} \times l$$

For simplicity:
path length l =1

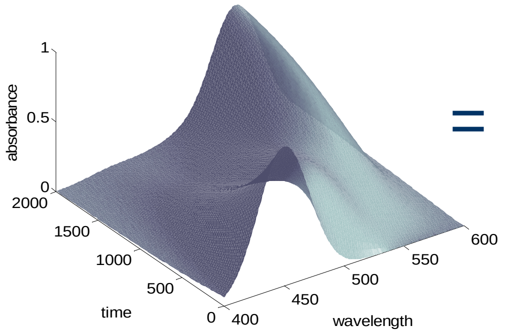
Beer’s law in elegant matrix notation



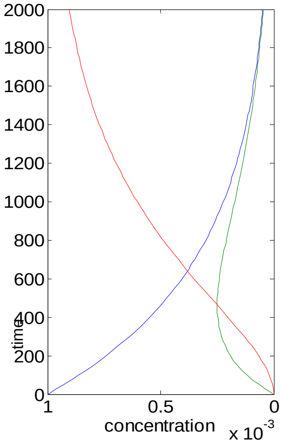
$$y_{t,\lambda} = \sum_{k=1}^{N_c} c_{t,k} a_{k,\lambda} + r_{t,\lambda}$$

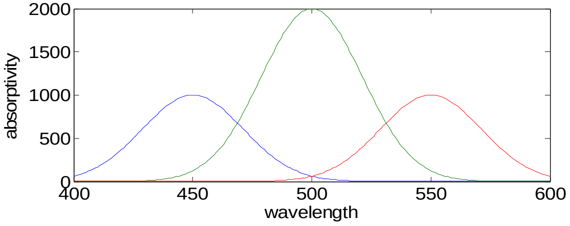
Rotational Ambiguity

e.g. A→B→C

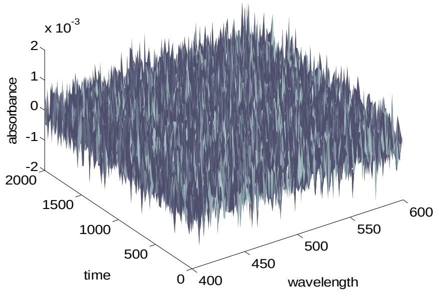


=





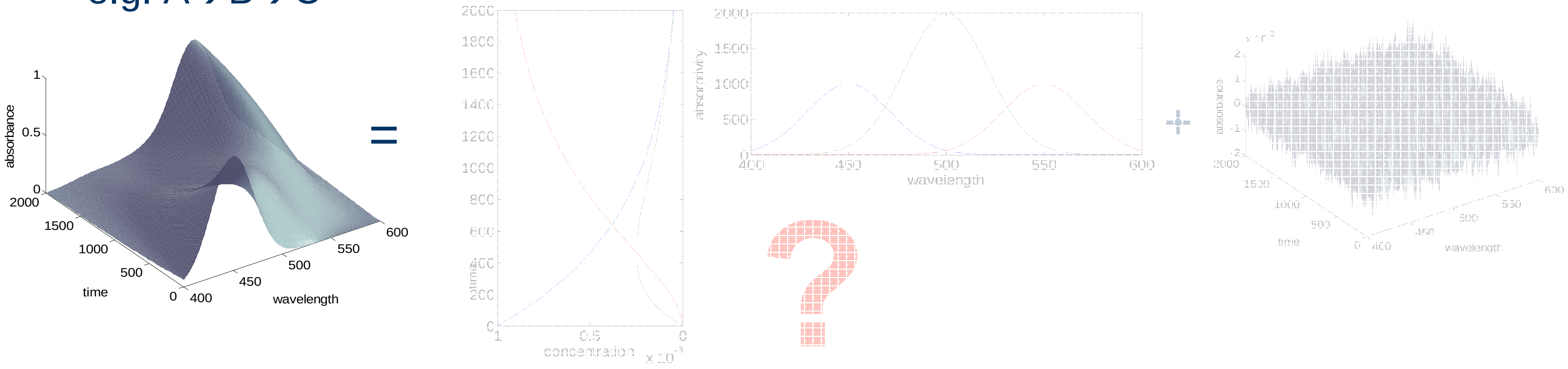
+



$$\mathbf{Y} = \mathbf{C} \times \mathbf{A} + \mathbf{R}$$

Rotational Ambiguity

e.g. A→B→C

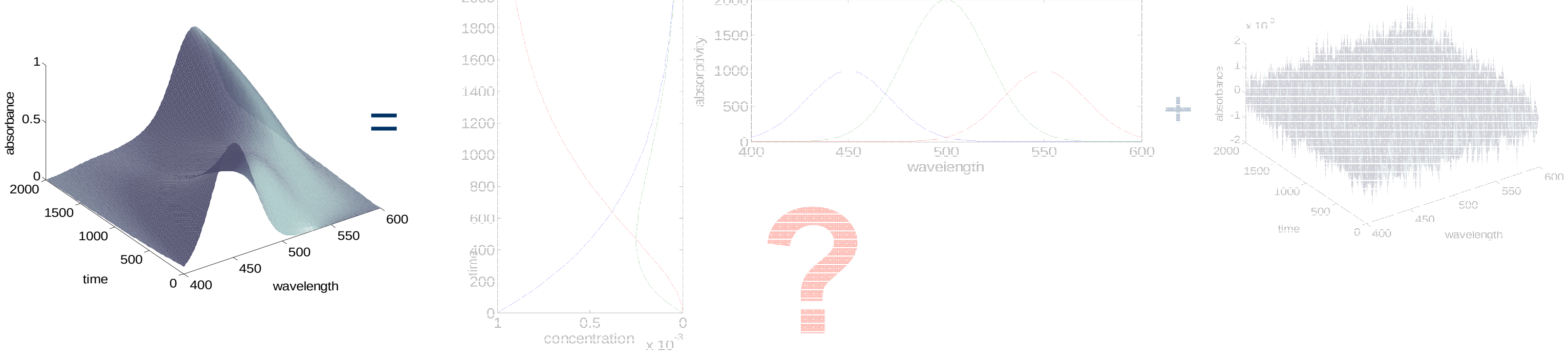


$$\mathbf{Y} = \mathbf{C} \times \mathbf{A} + \mathbf{R}$$

Goal: Find concentration profiles **C** and species spectra **A** such that the residuals **R**=**Y**-**CA** become small only using a ‘soft model’, i.e. by linear factorisation

Rotational Ambiguity

e.g. A→B→C



$$\mathbf{Y} \quad = \quad \mathbf{C} \quad \times \quad \mathbf{A} \quad + \quad \mathbf{R}$$

Goal: Find concentration profiles **C** and species spectra **A** such that the residuals **R**=**Y**-**CA** become small only using a ‘soft model’, i.e. by linear factorisation

Problem: Factorisation is not unique (rotational ambiguity)

Major Soft-Modelling Classes

- By using appropriate ‘soft’ restrictions on **C** and **A**, e.g. non-negativity, windows of existence, closure, unimodality, known spectra, the number of possible solutions can be reduced, sometimes can even lead to a unique solution for **C** & **A**

Major Soft-Modelling Classes

- By using appropriate ‘soft’ restrictions on **C** and **A**, e.g. non-negativity, windows of existence, closure, unimodality, known spectra, the number of possible solutions can be reduced, sometimes can even lead to a unique solution for **C** & **A**
- There are 2 major classes

Factor Analysis (AFA) based

$$\mathbf{Y} = \bar{\mathbf{U}}\bar{\mathbf{S}}\bar{\mathbf{V}} = \bar{\mathbf{U}}\mathbf{T}\mathbf{T}^{-1}\bar{\mathbf{S}}\bar{\mathbf{V}} = \mathbf{CA}$$

Find **T** such that

$$\mathbf{C} = \bar{\mathbf{U}}\mathbf{T}, \text{ and } \mathbf{A} = \mathbf{T}^{-1}\bar{\mathbf{S}}\bar{\mathbf{V}}$$

Alternating Least Squares (ALS) based

Start from some guessed **C**,

then recalculate **A** and **C** until satisfied:

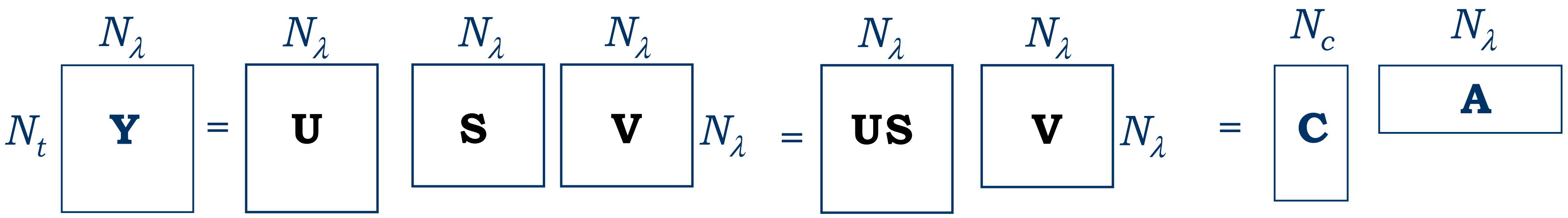
$$\mathbf{A} = (\mathbf{C}^t\mathbf{C})^{-1}\mathbf{C}^t\mathbf{Y} = \mathbf{C}^+\mathbf{Y}$$

$$\mathbf{C} = \mathbf{YA}^t(\mathbf{AA}^t)^{-1} = \mathbf{YA}^+$$

Principal Component Analysis (PCA)

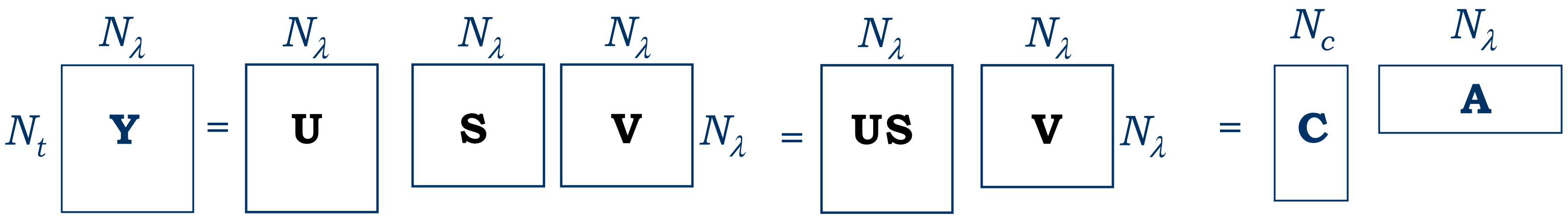
Principal Component Analysis (PCA)

- One very well defined solution is the one received from Abstract Factor Analysis (AFA) using Singular Value Decomposition (SVD)



Principal Component Analysis (PCA)

- One very well defined solution is the one received from Abstract Factor Analysis (AFA) using Singular Value Decomposition (SVD)

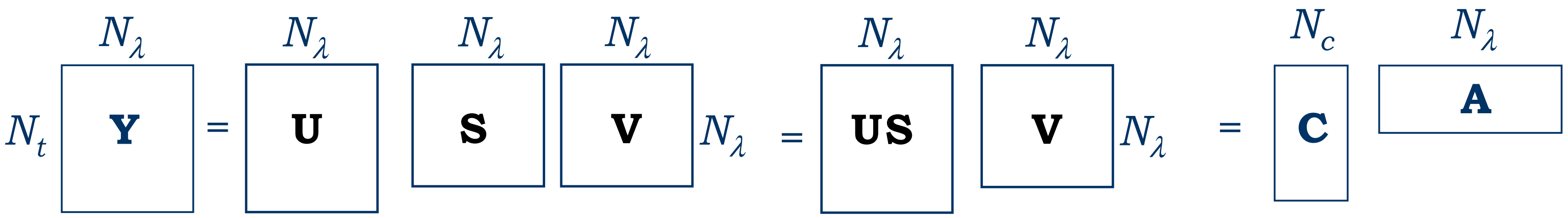


in Matlab:

```
[U,S,Vt]=svd(Y,0);
```

Principal Component Analysis (PCA)

- One very well defined solution is the one received from Abstract Factor Analysis (AFA) using Singular Value Decomposition (SVD)



in Matlab:

```
[U,S,Vt]=svd(Y,0);
```

- Some properties of **U**, **S** and **V**

$$\mathbf{YY}^t\mathbf{U} = \mathbf{U\Lambda}$$

$$\mathbf{Y}^t\mathbf{YV}^t = \mathbf{V}^t\mathbf{\Lambda}$$

$$\mathbf{\Lambda} = \mathbf{S}^2$$

$$\mathbf{U}^t\mathbf{U} = \mathbf{VV}^t = \mathbf{I}$$

columns of **U** (rows of **V**) are eigenvectors of **YY**^{t} (**Y**^{t}**Y**)

S is a diagonal matrix with the square root of their eigenvalues

U and **V**^{t} are orthonormal

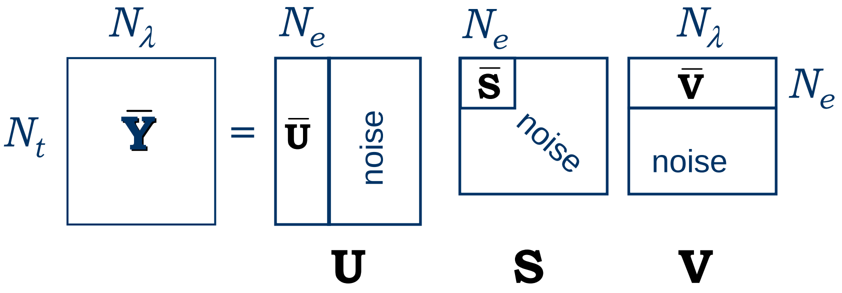
Chemical rank – number of absorbing species

Chemical rank – number of absorbing species

- Eigenvectors in **U** (columns) and **V** (rows) are arranged in decreasing order of magnitude of their corresponding singular values in **S**

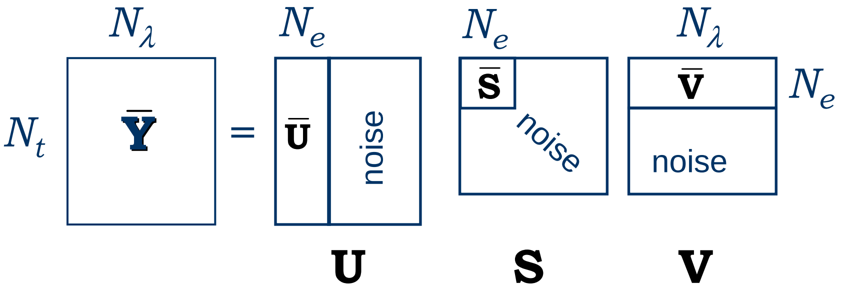
Chemical rank – number of absorbing species

- Eigenvectors in **U** (columns) and **V** (rows) are arranged in decreasing order of magnitude of their corresponding singular values in **S**
- Many of them just represent ‘noise’ and can be neglected; the significant ‘factors’, the Principal Components, are retained in $\bar{\mathbf{U}}$ and $\bar{\mathbf{V}}$ and form ‘abstract’ concentration profiles and spectra



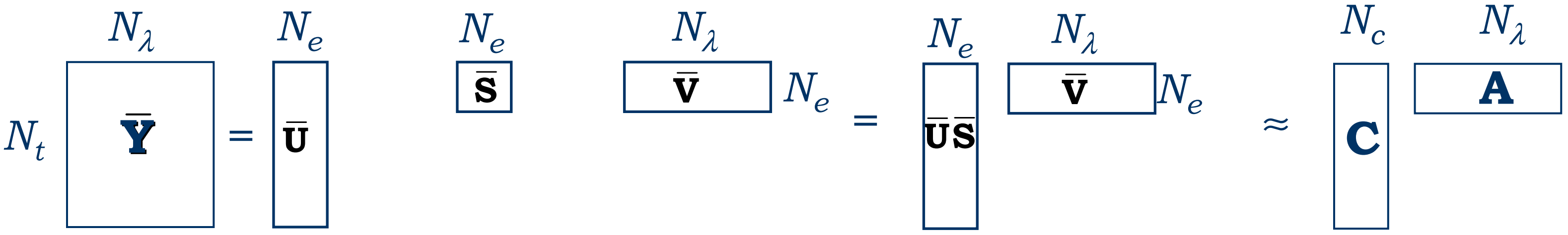
Chemical rank – number of absorbing species

- Eigenvectors in **U** (columns) and **V** (rows) are arranged in decreasing order of magnitude of their corresponding singular values in **S**
- Many of them just represent ‘noise’ and can be neglected; the significant ‘factors’, the Principal Components, are retained in $\bar{\mathbf{U}}$ and $\bar{\mathbf{V}}$ and form ‘abstract’ concentration profiles and spectra
- The diagonal elements of $\bar{\mathbf{S}}$, the singular values, can be seen as normalisation coefficients for $\bar{\mathbf{U}}$ or $\bar{\mathbf{V}}$



Chemical rank – number of absorbing species

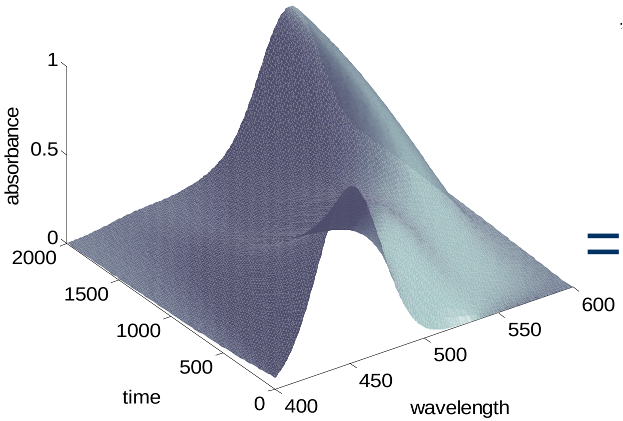
- Eigenvectors in **U** (columns) and **V** (rows) are arranged in decreasing order of magnitude of their corresponding singular values in **S**
- Many of them just represent ‘noise’ and can be neglected; the significant ‘factors’, the Principal Components, are retained in $\bar{\mathbf{U}}$ and $\bar{\mathbf{V}}$ and form ‘abstract’ concentration profiles and spectra
- The diagonal elements of $\bar{\mathbf{S}}$, the singular values, can be seen as normalisation coefficients for $\bar{\mathbf{U}}$ or $\bar{\mathbf{V}}$



- The number of significant singular(eigen) values and –vectors is the chemical rank of **Y** and a 1st estimate on the number of absorbing species

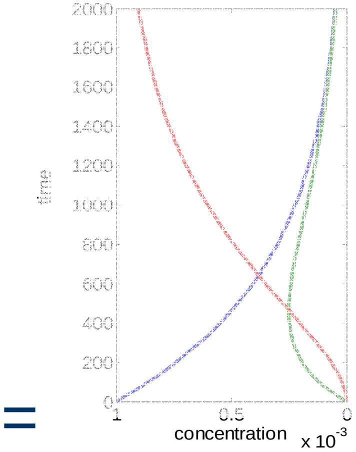
Chemical rank – number of absorbing species

e.g. A→B→C

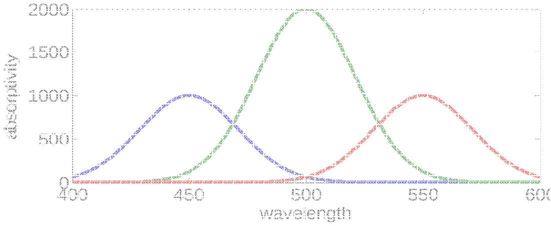


Y

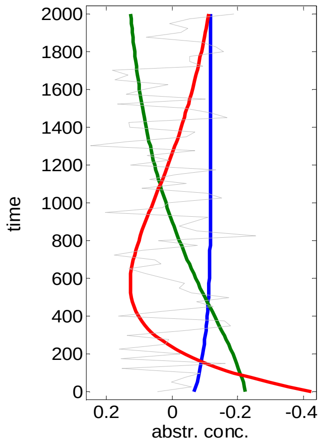
=



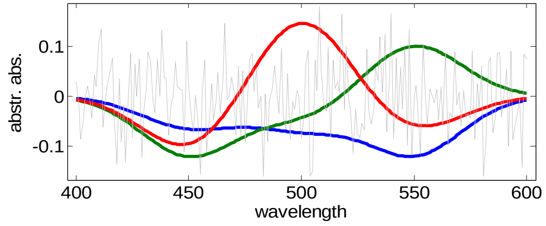
C



A + **R**



$$\begin{bmatrix} 43 & & 0 \\ & 22 & \\ 0 & & 6 \end{bmatrix} 10^{-2}$$



$+ \mathbf{R}_{PCA}$

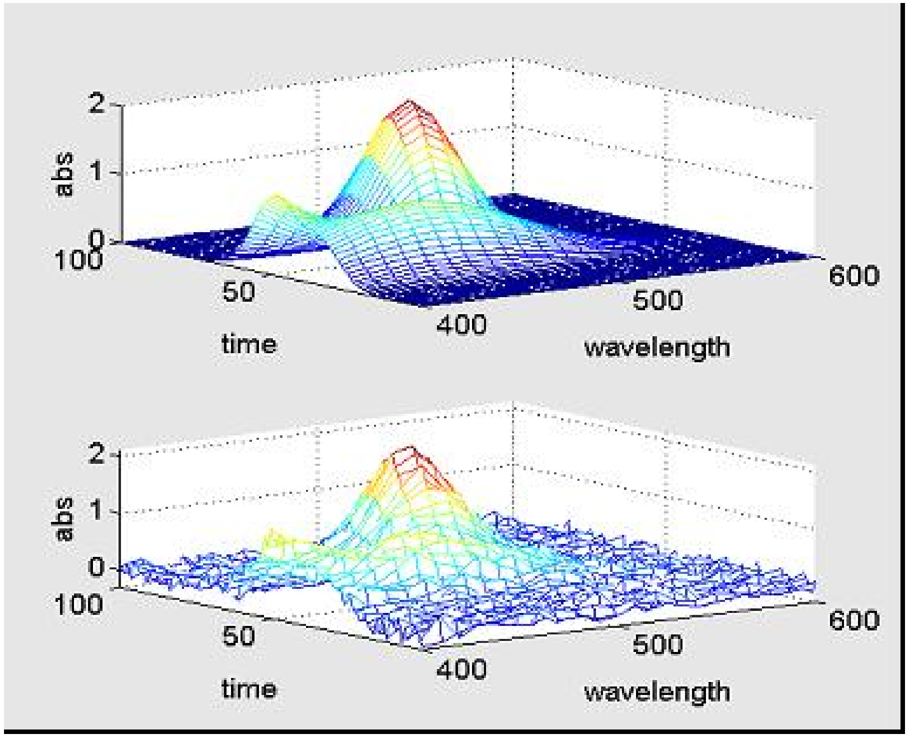
$\bar{\mathbf{U}}$ $\bar{\mathbf{S}}$ $\bar{\mathbf{V}}$

→ rank(**Y**) = 3

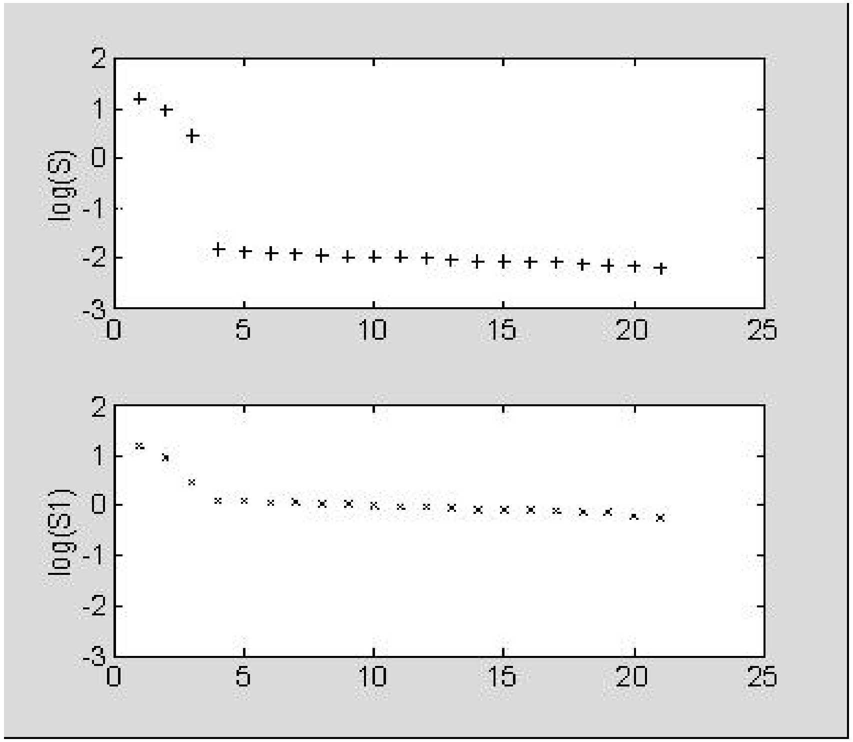
→ 3 absorbing species

Chemical rank – number of absorbing species

Y ($N_t \times N_\lambda$)

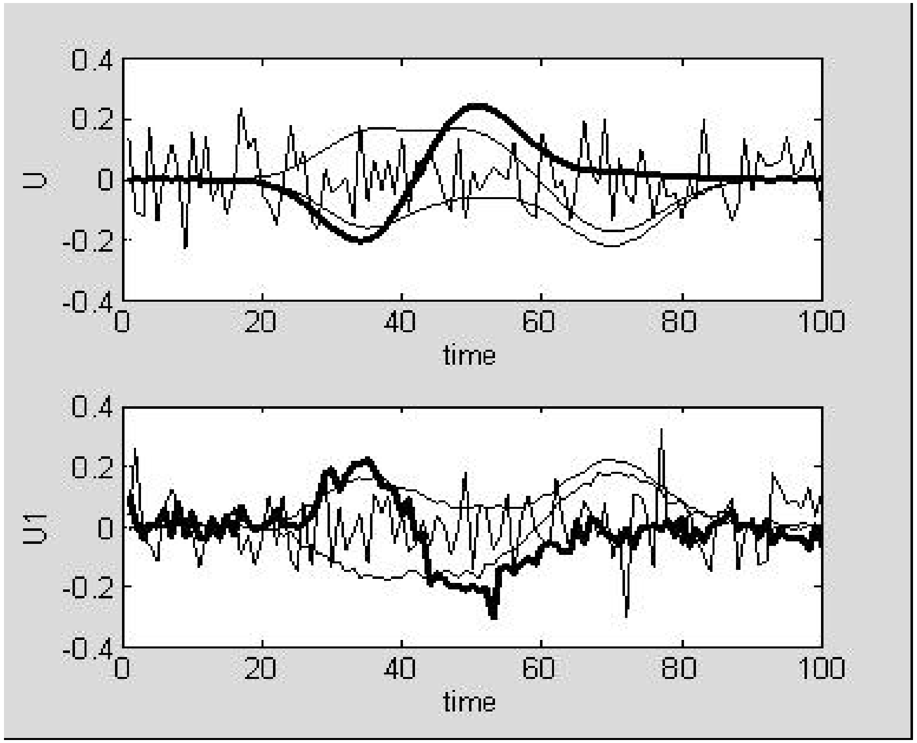


log($s_{i,i}$) vs i



The noise level in the data matrix **Y** determines the drop in the magnitude from significant to insignificant singular values

Chemical rank – number of absorbing species



The noise level in the data matrix **Y** also determines the remaining noise in the significant singular vectors

The signs of the singular vectors can interchange between **U** and **V**)

Geometric interpretations

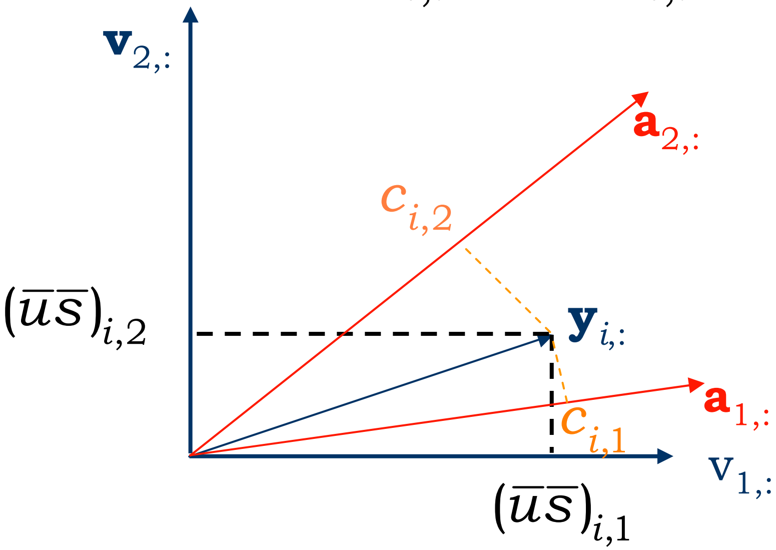
Geometric interpretations

- e.g. 2 species, A→ B
- the (significant) eigenvectors $\mathbf{v}_{1,:}$ and $\mathbf{v}_{2,:}$ form an orthonormal base in the same ‘plane’ as the pure species spectra $\mathbf{a}_{1,:}$ and $\mathbf{a}_{2,:}$

Geometric interpretations

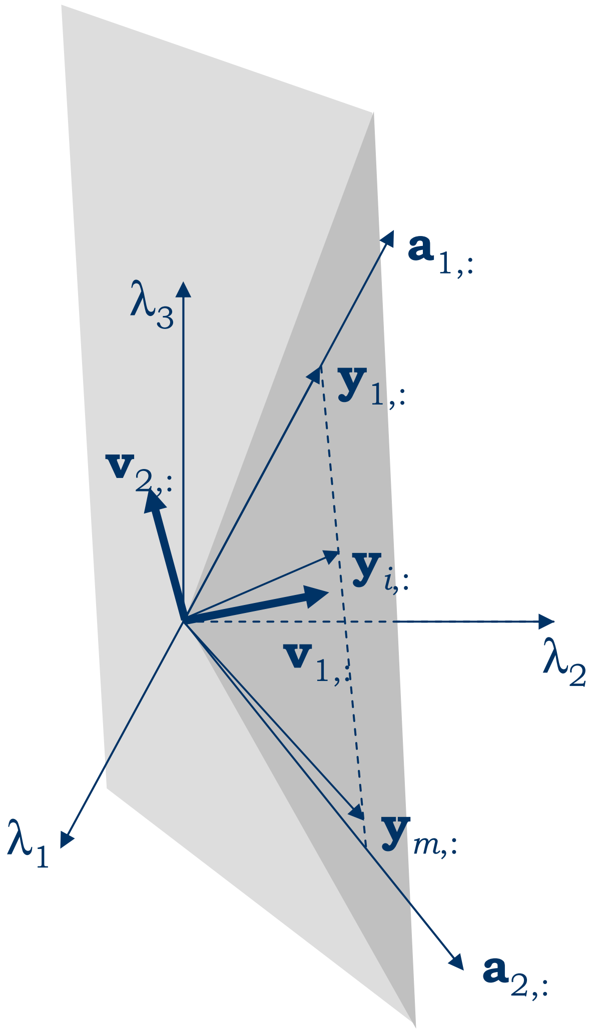
- e.g. 2 species, A→ B
- the (significant) eigenvectors $\mathbf{v}_{1,:}$ and $\mathbf{v}_{2,:}$ form an orthonormal base in the same ‘plane’ as the pure species spectra $\mathbf{a}_{1,:}$ and $\mathbf{a}_{2,:}$

$$\mathbf{y}_{i,:} \approx (\overline{\mathbf{u}}\overline{\mathbf{s}})_{i,:}\overline{\mathbf{V}} \approx \mathbf{c}_{i,:}\mathbf{A}$$

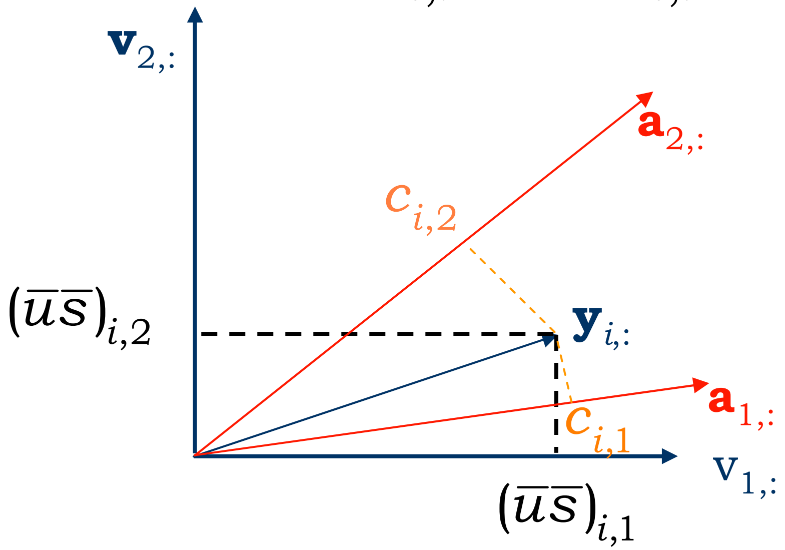


Geometric interpretations

- e.g. 2 species, A→ B
- the (significant) eigenvectors $\mathbf{v}_{1,:}$ and $\mathbf{v}_{2,:}$ form an orthonormal base in the same ‘plane’ as the pure species spectra $\mathbf{a}_{1,:}$ and $\mathbf{a}_{2,:}$



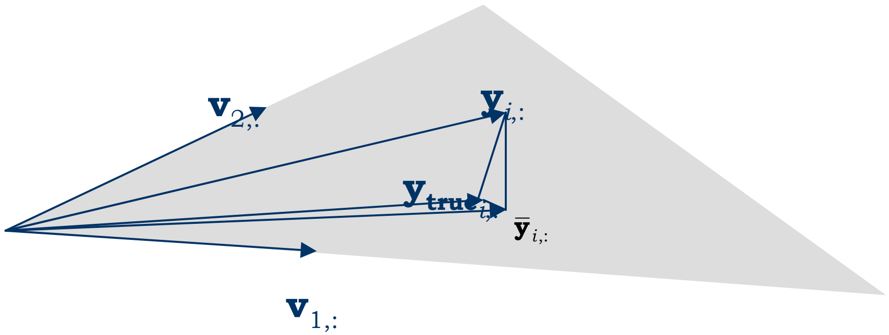
$$\mathbf{y}_{i,:} \approx (\overline{\mathbf{u}}\overline{\mathbf{s}})_{i,:}\overline{\mathbf{V}} \approx \mathbf{c}_{i,:}\mathbf{A}$$

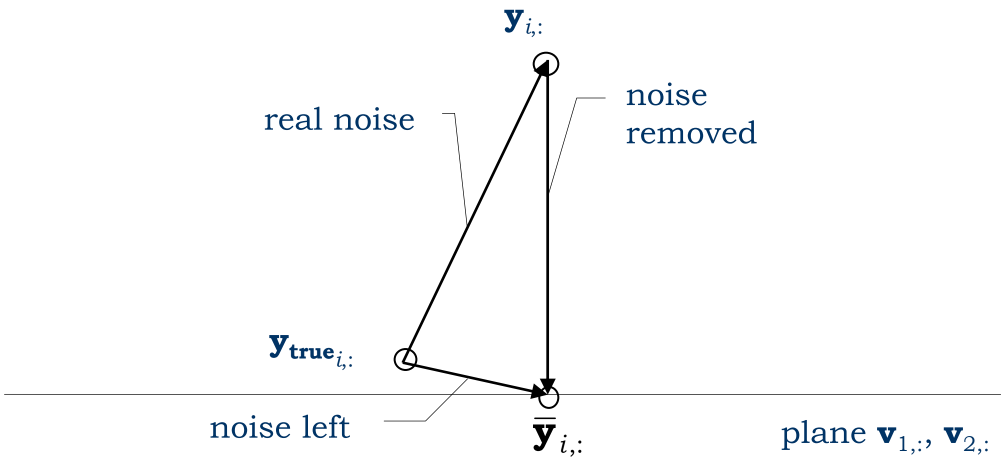


Some noise reduction

Some noise reduction

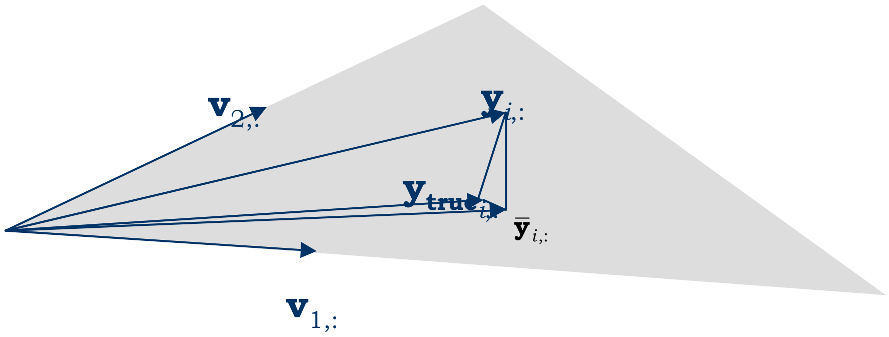
$$\mathbf{Y} = \bar{\mathbf{Y}} + \mathbf{R}_{PCA} = \sum_{i=1}^{N_e} \mathbf{u}_{:,i} \mathbf{s}_{i,i} \mathbf{v}_{i,:} + \sum_{j=N_e+1}^{N_\lambda} \mathbf{u}_{:,j} \mathbf{s}_{j,j} \mathbf{v}_{j,:} = \sum_{k=1}^{N_c} \mathbf{c}_{:,k} \mathbf{a}_{k,:} + \mathbf{R}_{noise}$$

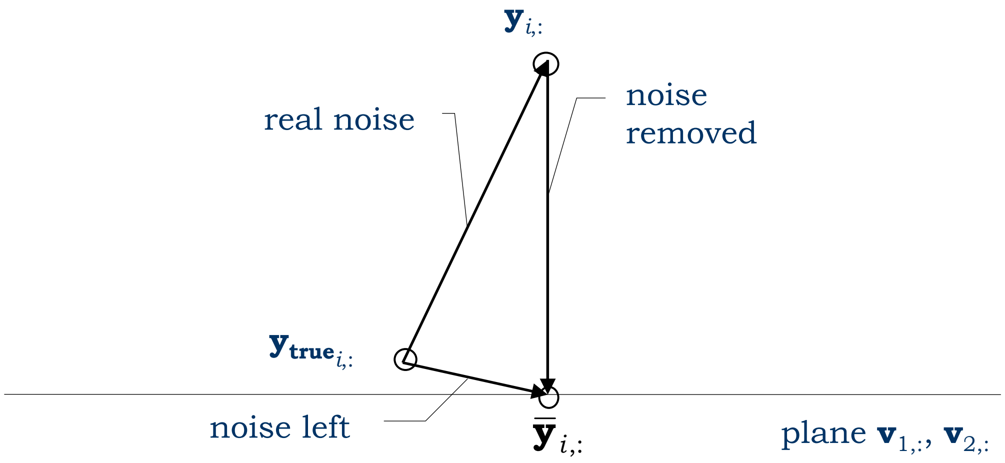


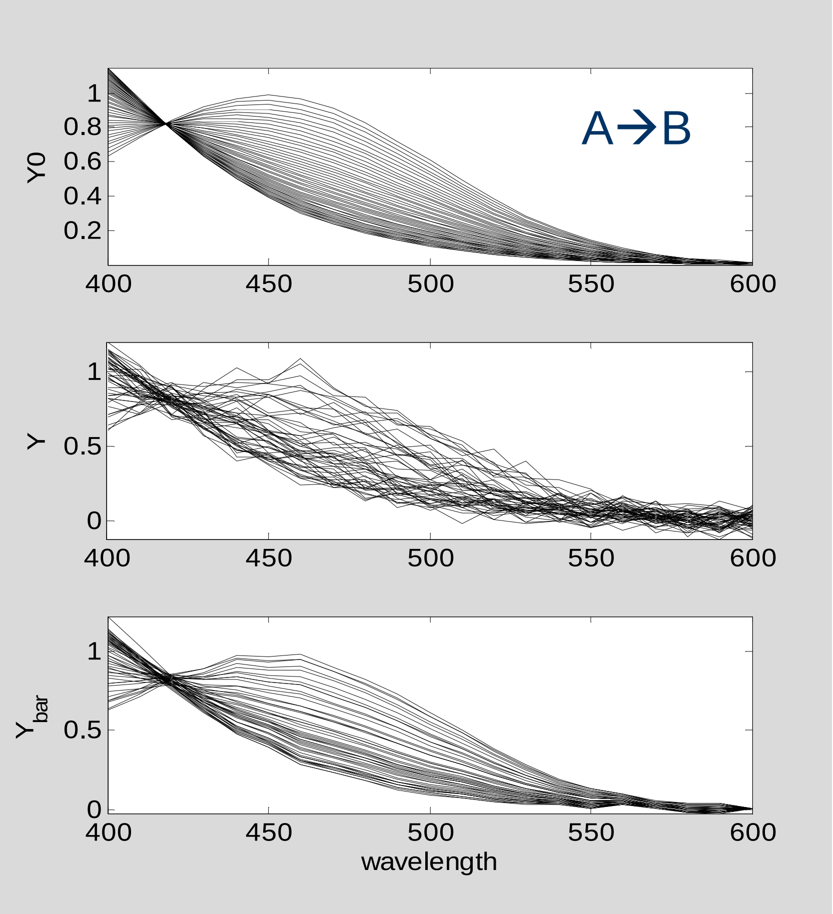

$\mathbf{y}_{i,:}$
real noise
noise removed
$\mathbf{y}_{\mathbf{true}_{i,:}}$
noise left
$\bar{\mathbf{y}}_{i,:}$
plane $\mathbf{v}_{1,:}$, $\mathbf{v}_{2,:}$

Some noise reduction

$$\mathbf{Y} = \bar{\mathbf{Y}} + \mathbf{R}_{PCA} = \sum_{i=1}^{N_e} \mathbf{u}_{:,i}\mathbf{s}_{i,i}\mathbf{v}_{i,:} + \sum_{j=N_e+1}^{N_\lambda} \mathbf{u}_{:,j}\mathbf{s}_{j,j}\mathbf{v}_{j,:} = \sum_{k=1}^{N_c} \mathbf{c}_{:,k}\mathbf{a}_{k,:} + \mathbf{R}_{noise}$$







More precise statements on the chemical rank

More precise statements on the chemical rank

- the chemical rank of the spectral data matrix **Y** is determined by the number of its significant singular vectors

More precise statements on the chemical rank

- the chemical rank of the spectral data matrix **Y** is determined by the number of its significant singular vectors
- the number of significant singular vectors of **Y** is determined by the number of linearly independent columns or rows in the matrix of pure species spectra (**A**) and corresponding concentration profiles (**C**)

More precise statements on the chemical rank

- the chemical rank of the spectral data matrix **Y** is determined by the number of its significant singular vectors
- the number of significant singular vectors of **Y** is determined by the number of linearly independent columns or rows in the matrix of pure species spectra (**A**) and corresponding concentration profiles (**C**)
- linear dependencies in **C** due to the kinetic model are common and sometimes difficult to predict (e.g. $A+B \rightarrow C$)

More precise statements on the chemical rank

- the chemical rank of the spectral data matrix **Y** is determined by the number of its significant singular vectors
- the number of significant singular vectors of **Y** is determined by the number of linearly independent columns or rows in the matrix of pure species spectra (**A**) and corresponding concentration profiles (**C**)
- linear dependencies in **C** due to the kinetic model are common and sometimes difficult to predict (e.g. *A*+*B*→*C*)
- linear dependencies in **A** are less common

Evolving Factor Analysis (EFA)

Evolving Factor Analysis (EFA)

- sequential rank analysis of the data matrix along its time domain by repeated SVD

Evolving Factor Analysis (EFA)

- sequential rank analysis of the data matrix along its time domain by repeated SVD
- can be performed in a forward and backward way

Evolving Factor Analysis (EFA)

- sequential rank analysis of the data matrix along its time domain by repeated SVD
- can be performed in a forward and backward way
- indicates the rise of new singular vectors and thus gives an estimate for the appearance & disappearance of new absorbing species

Evolving Factor Analysis (EFA)

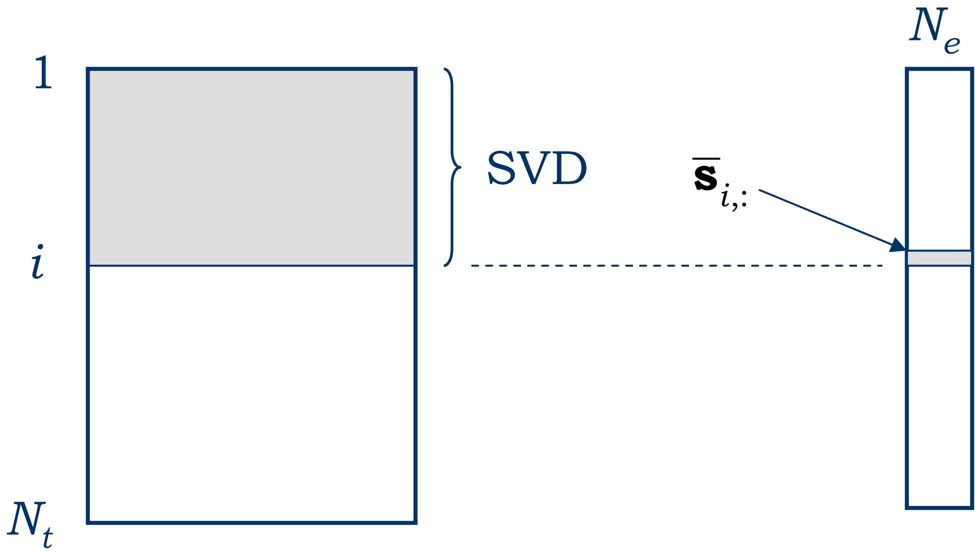
- sequential rank analysis of the data matrix along its time domain by repeated SVD
- can be performed in a forward and backward way
- indicates the rise of new singular vectors and thus gives an estimate for the appearance & disappearance of new absorbing species
- ideally designed to follow chromatography experiments
  - species appear & disappear sequentially

Evolving Factor Analysis (EFA)

- sequential rank analysis of the data matrix along its time domain by repeated SVD
- can be performed in a forward and backward way
- indicates the rise of new singular vectors and thus gives an estimate for the appearance & disappearance of new absorbing species
- ideally designed to follow chromatography experiments
  - species appear & disappear sequentially
- capable of roughly following kinetic profiles
  - species can appear & dissappear simultaneously

Evolving Factor Analysis (EFA)

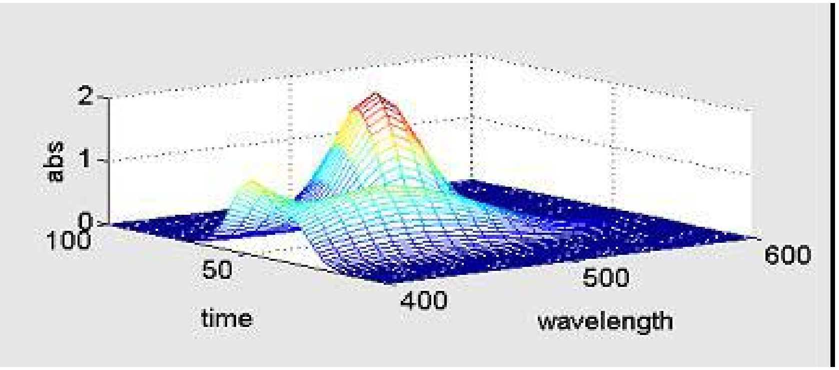
Forward EFA



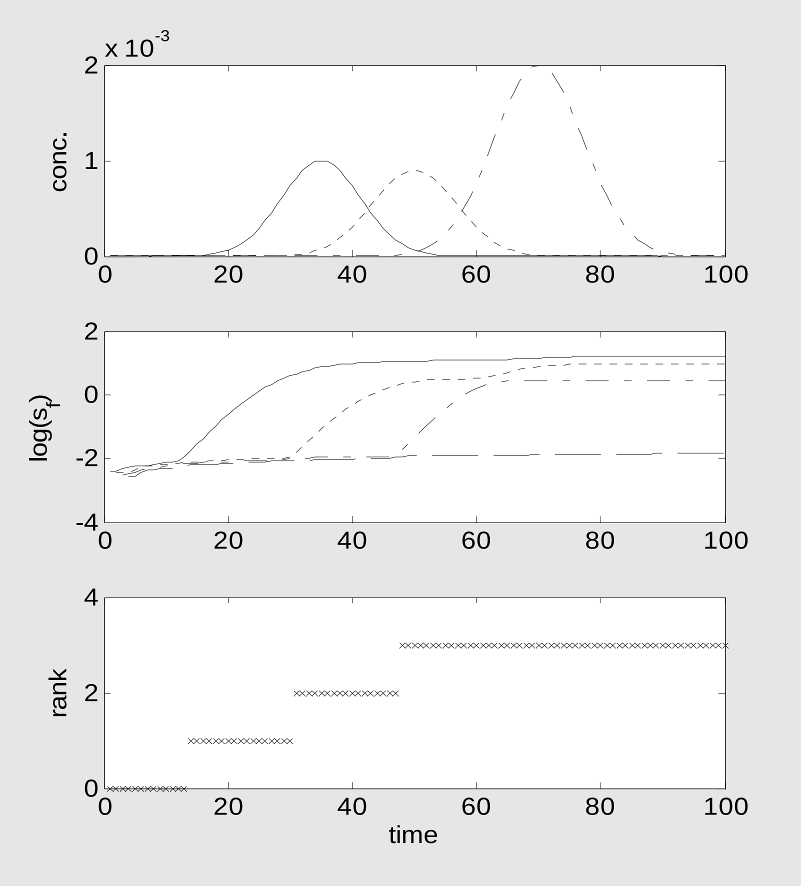
- Repeated rank analysis by SVD in forward direction
- The appearance of a new ‘species’ is indicated by a gradual rise of a new singular value

Evolving Factor Analysis (EFA)

Chromatography

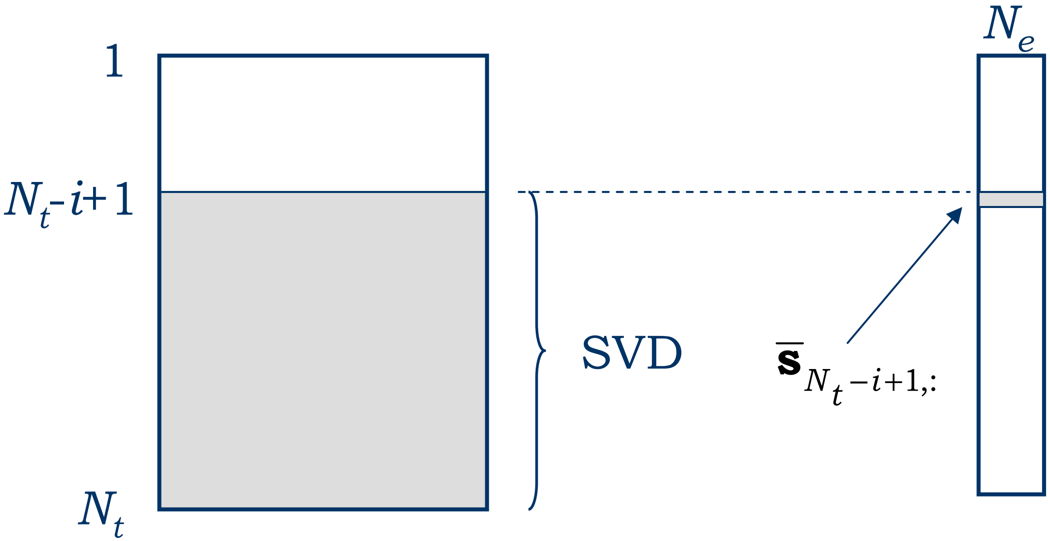


Forward EFA



Evolving Factor Analysis (EFA)

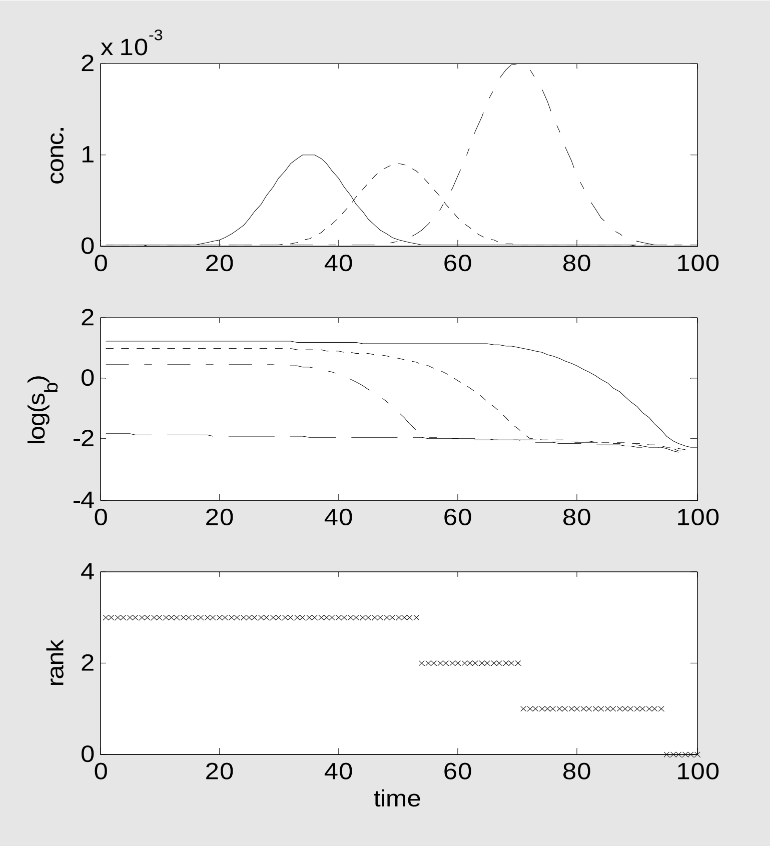
Backward EFA



- Repeated rank analysis by SVD in backward direction
- A ‘disappearing species’ is indicated by a gradual rise of a new singular value

Evolving Factor Analysis (EFA)

Backward EFA



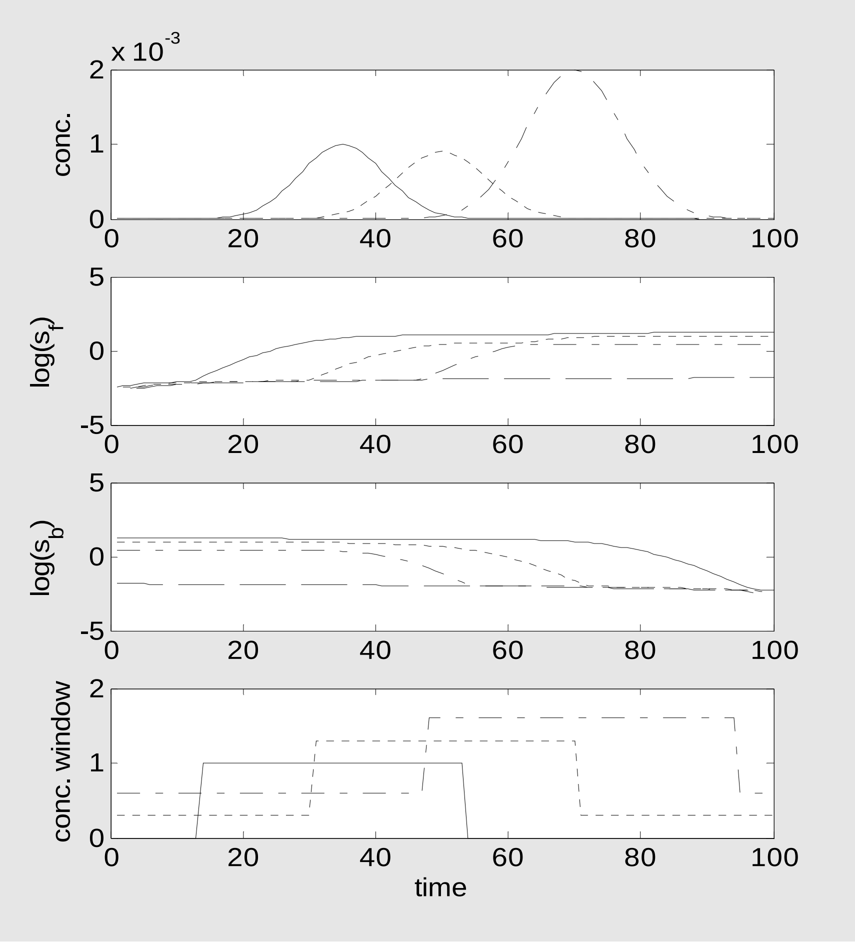
Matlab script using a function for forward/backward EFA

```
[t,lam,Y,C,A]=Data_Chrom2;
[ns,nc]=size(C);
ne=nc+1;               % one extra sing. val.
[EFA_f,EFA_b]=EFA(Y,ne);
```

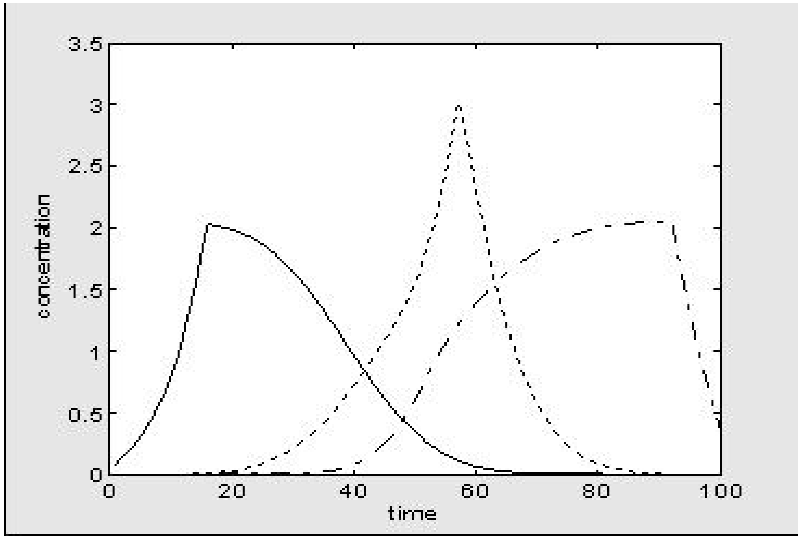
```
function [EFA_f,EFA_b]=EFA(Y,ne)
[ns,nl]=size(Y);
EFA_f=NaN(ns,ne);
EFA_b=NaN(ns,ne);
for i=1:ns
   s_f=svd(Y(1:i,:));                  % forward sv
   s_b=svd(Y(ns-i+1:ns,:));            % backward sv
   EFA_f(i,1:min(i,ne))=s_f(1:min(i,ne))';
   EFA_b(ns-i+1,1:min(i,ne))=s_b(1:min(i,ne))';
end
```

Evolving Factor Analysis (EFA)

Forward and backward EFA



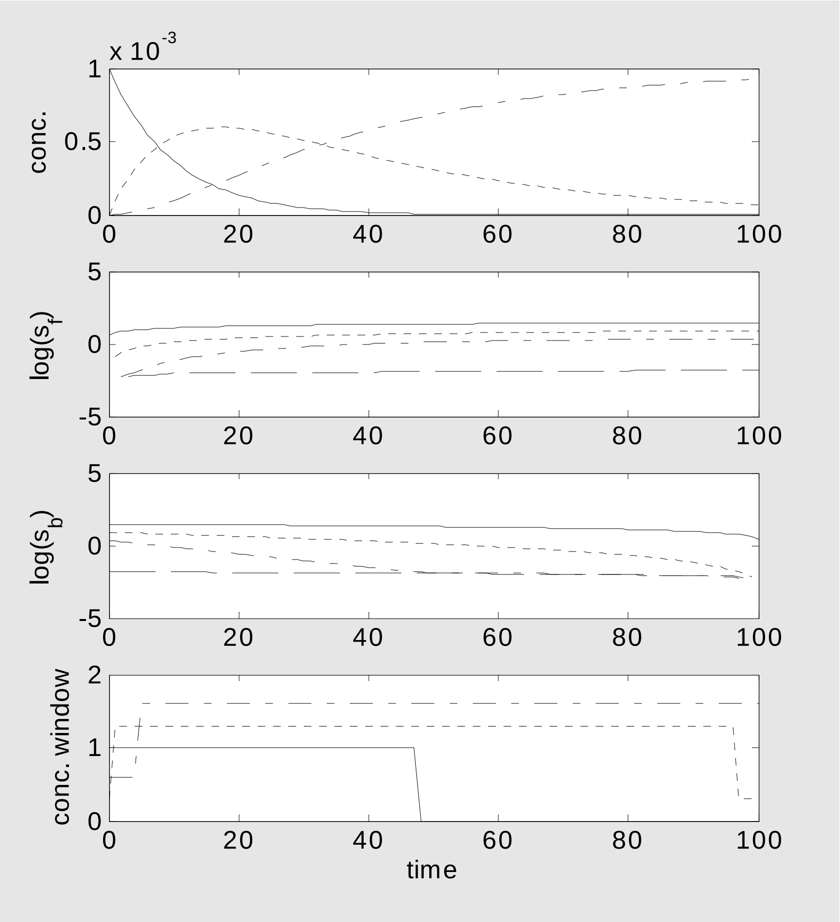
Combined forward/backward EFA results can be used as reasonable initial guesses of concentration profiles for subsequent iterative refinement e.g. by ALS



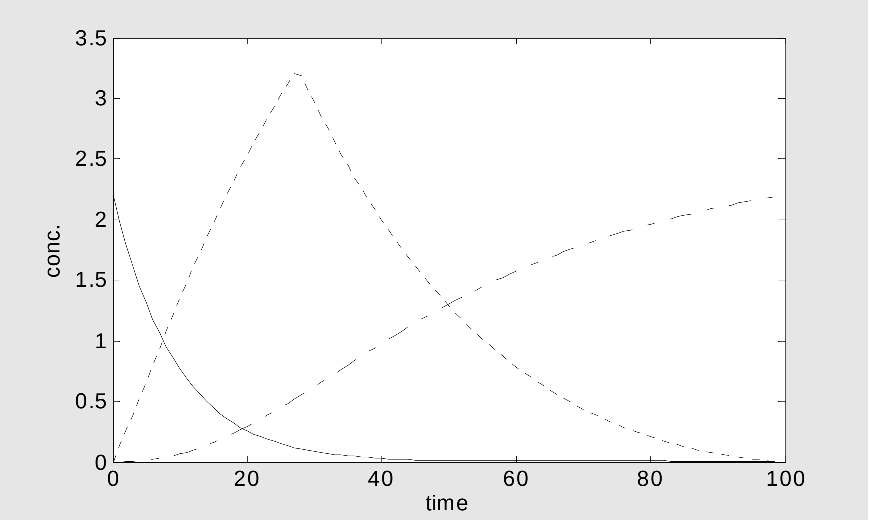
```
% combined SV curves
C=min(EFA_f(:,1:nc),fliplr(EFA_b(:,1:nc)));
```

Evolving Factor Analysis (EFA)

A→B→C, Forward and backward EFA

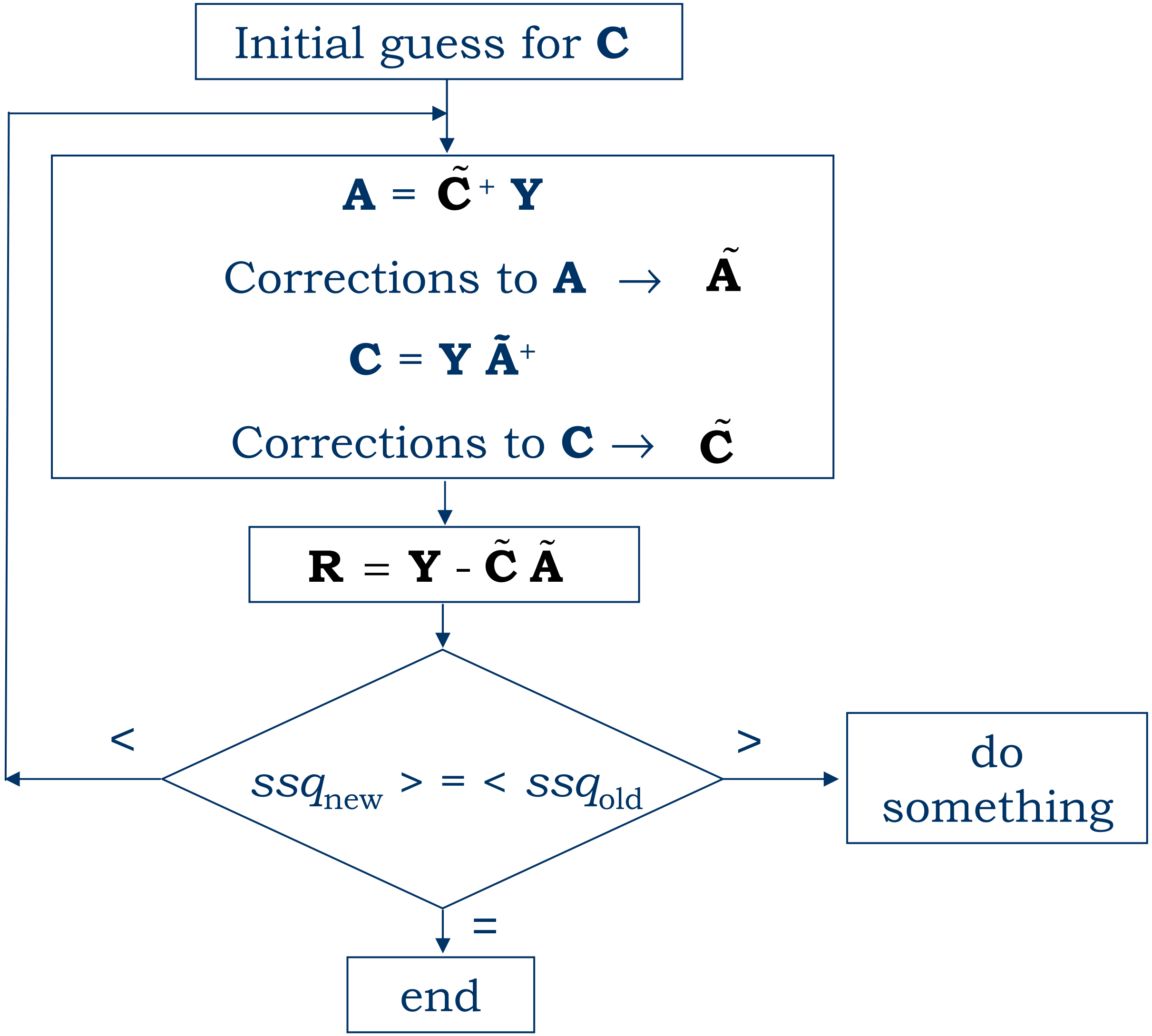


- Combined forward/backward EFA results are not as accurate as in chromatography regarding the appearance and disappearance of species
- But they can still be used as initial guesses of concentration profiles for subsequent iterative refinement e.g. by ALS



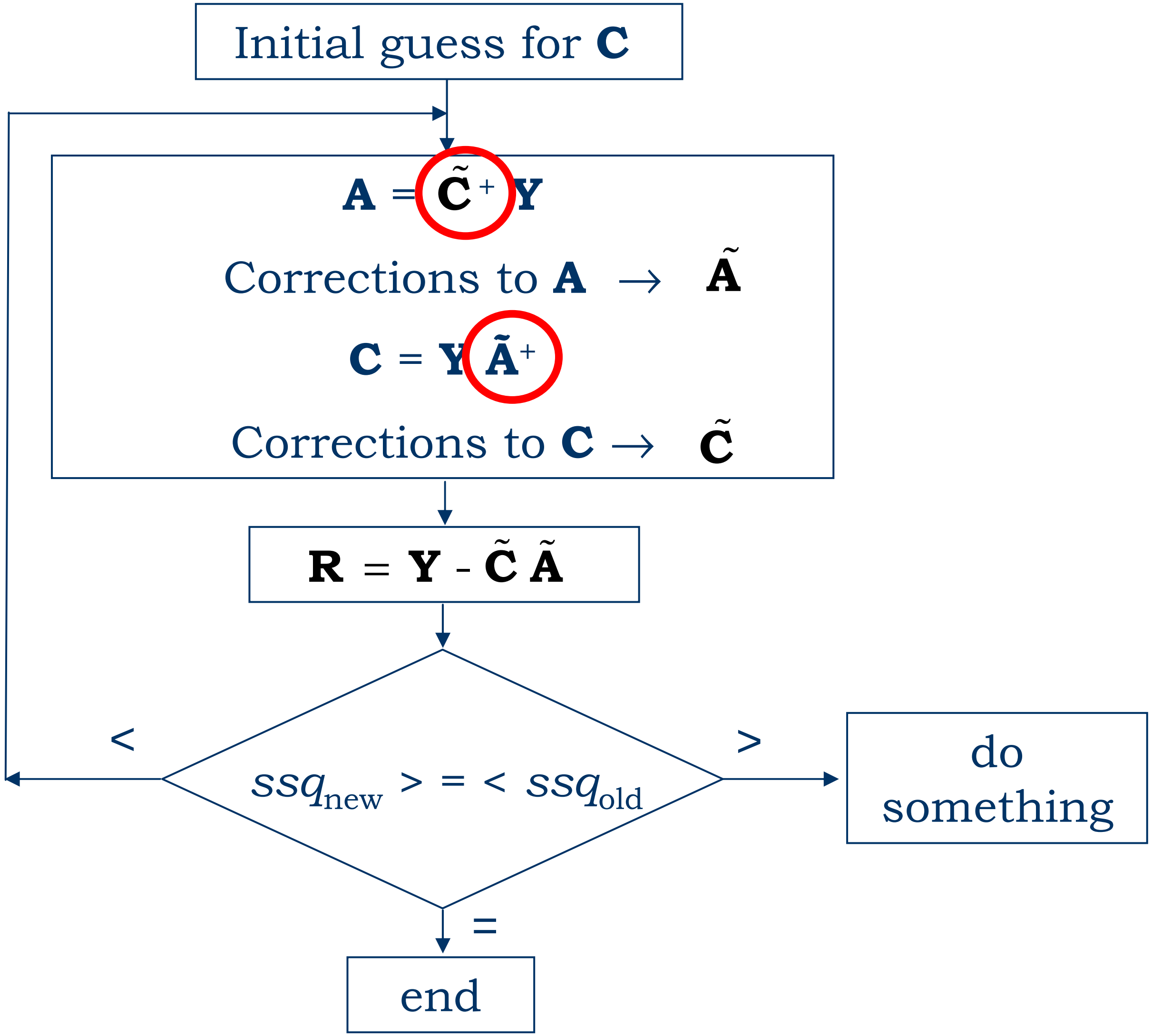
Multivariate Curve Resolution by Alternating Least-Squares (MCR-ALS)

- conceptually very simple


Initial guess for **C**
$\mathbf{A} = \tilde{\mathbf{C}}^{+}\,\mathbf{Y}$
Corrections to **A** → $\tilde{\mathbf{A}}$
$\mathbf{C} = \mathbf{Y}\,\tilde{\mathbf{A}}^{+}$
Corrections to **C** → $\tilde{\mathbf{C}}$
$\mathbf{R} = \mathbf{Y} - \tilde{\mathbf{C}}\,\tilde{\mathbf{A}}$
$ssq_{new} > = < ssq_{old}$
<
>
=
do something
end

Multivariate Curve Resolution by Alternating Least-Squares (MCR-ALS)

- conceptually very simple



Initial guess for **C**

$\mathbf{A} = \tilde{\mathbf{C}}^{+}\mathbf{Y}$

Corrections to **A** → $\tilde{\mathbf{A}}$

$\mathbf{C} = \mathbf{Y}\tilde{\mathbf{A}}^{+}$

Corrections to **C** → $\tilde{\mathbf{C}}$

$\mathbf{R} = \mathbf{Y} - \tilde{\mathbf{C}}\tilde{\mathbf{A}}$

<

$ssq_{new} > = < ssq_{old}$

>

do something

=

end

The pseudo inverse and the linear least-squares solution

The pseudo inverse and the linear least-squares solution

$$\mathbf{Y} = \mathbf{C} \times \mathbf{A} + \mathbf{R}$$

$$\mathbf{R} = \mathbf{Y} - \mathbf{C} \times \mathbf{A}$$

$$\mathbf{R} := f(\mathbf{Y},\ \mathbf{C},\ \mathbf{A})$$

The residuals **R** are a function of **Y** and the two linear parameters **C** & **A**

The pseudo inverse and the linear least-squares solution

$$\mathbf{Y} = \mathbf{C} \times \mathbf{A} + \mathbf{R}$$

$$\mathbf{R} = \mathbf{Y} - \mathbf{C} \times \mathbf{A}$$

$$\mathbf{R} := f(\mathbf{Y}, \mathbf{C}, \mathbf{A})$$

The residuals **R** are a function of **Y** and the two linear parameters **C** & **A**

1) **Y** & **C** known: $min\left\|\Sigma\Sigma r_{i,j}^2(\mathbf{A})\right\|$

R is minimal in the least squares sense if

$$\mathbf{A} = (\mathbf{C}^{\mathrm{t}}\mathbf{C})^{-1}\mathbf{C}^{\mathrm{t}} \times \mathbf{Y} = \mathbf{C}^{+} \times \mathbf{Y}$$

The pseudo inverse and the linear least-squares solution

$$\mathbf{Y} = \mathbf{C} \times \mathbf{A} + \mathbf{R}$$

$$\mathbf{R} = \mathbf{Y} - \mathbf{C} \times \mathbf{A}$$

$$\mathbf{R} := f(\mathbf{Y},\ \mathbf{C},\ \mathbf{A})$$

The residuals **R** are a function of **Y** and the two linear parameters **C** & **A**

1) **Y** & **C** known: $min \left\| \Sigma\Sigma r_{i,j}^{2}(\mathbf{A}) \right\|$

R is minimal in the least squares sense if

$$\mathbf{A} = (\mathbf{C}^{t}\mathbf{C})^{-1}\mathbf{C}^{t} \times \mathbf{Y} = \mathbf{C}^{+} \times \mathbf{Y}$$

```
Matlab: A = C \ Y
```

to use the left pseudo inverse of **C**

The pseudo inverse and the linear least-squares solution

$$\mathbf{Y} = \mathbf{C}\times\mathbf{A}+\mathbf{R}$$

$$\mathbf{R} = \mathbf{Y}-\mathbf{C}\times\mathbf{A}$$

$$\mathbf{R} := f(\mathbf{Y},\ \mathbf{C},\ \mathbf{A})$$

The residuals **R** are a function of **Y** and the two linear parameters **C** & **A**

1) **Y** & **C** known: $min\left\|\Sigma\Sigma r_{i,j}^{2}(\mathbf{A})\right\|$

R is minimal in the least squares sense if

$$\mathbf{A} = (\mathbf{C}^{\mathrm{t}}\mathbf{C})^{-1}\mathbf{C}^{\mathrm{t}}\times\mathbf{Y} = \mathbf{C}^{+}\times\mathbf{Y}$$

`Matlab: A = C \ Y`

to use the left pseudo inverse of **C**

2) **Y** & **A** known: $min\left\|\Sigma\Sigma r_{i,j}^{2}(\mathbf{C})\right\|$

R is minimal in the least squares sense if

$$\mathbf{C} = \mathbf{Y}\times\mathbf{A}^{\mathrm{t}}(\mathbf{A}\mathbf{A}^{\mathrm{t}})^{-1} = \mathbf{Y}\times\mathbf{A}^{+}$$

The pseudo inverse and the linear least-squares solution

$$\mathbf{Y}=\mathbf{C}\times\mathbf{A}+\mathbf{R}$$

$$\mathbf{R}=\mathbf{Y}-\mathbf{C}\times\mathbf{A}$$

$$\mathbf{R}:=f(\mathbf{Y},\ \mathbf{C},\ \mathbf{A})$$

The residuals **R** are a function of **Y** and the two linear parameters **C** & **A**

1) Y & C known: $min\left\|\Sigma\Sigma r_{i,j}^{2}(\mathbf{A})\right\|$

R is minimal in the least squares sense if

$$\mathbf{A}=(\mathbf{C}^{t}\mathbf{C})^{-1}\mathbf{C}^{t}\times\mathbf{Y}=\mathbf{C}^{+}\times\mathbf{Y}$$

`Matlab: A = C \ Y`

to use the left pseudo inverse of C

2) **Y** & **A** known: $min\left\|\Sigma\Sigma r_{i,j}^{2}(\mathbf{C})\right\|$

R is minimal in the least squares sense if

$$\mathbf{C}=\mathbf{Y}\times\mathbf{A}^{t}(\mathbf{A}\mathbf{A}^{t})^{-1}=\mathbf{Y}\times\mathbf{A}^{+}$$

`Matlab: C = Y / A`

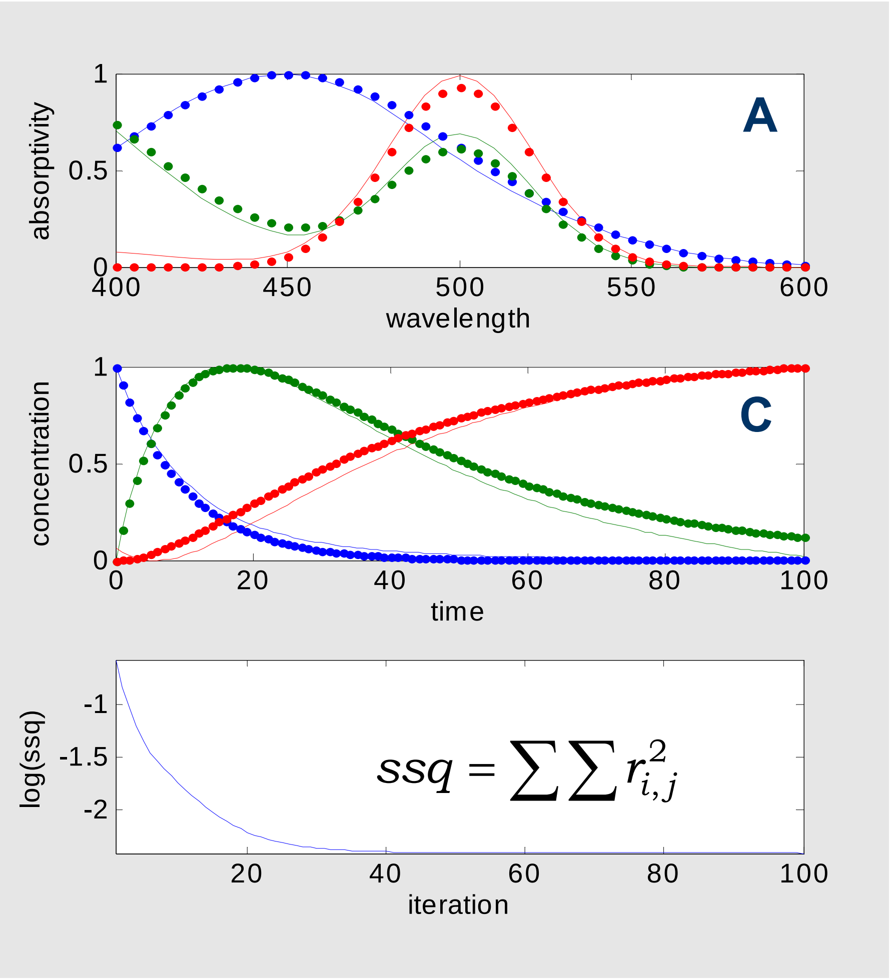
to use the left pseudo inverse of **C**

Multivariate Curve Resolution by Alternating Least-Squares (MCR-ALS)

```
for it=1:100
    C=norm_max(C);           % normalisation

    [C,A]=constraints_nonneg(Y,C);

    R=Y-C*A;                 % residuals
    ssq(it)=sum(sum(R.*R));
end
```

A→B→C

constraints: **C**, **A** >0



Normalisation to the maximum in each conc. profile

$$\boxed{\mathbf{C}_n} = \boxed{\mathbf{C}} \times \boxed{}$$

diag(max(**C**))$^{-1}$

and

$$\mathbf{A}_n = diag(\max(\mathbf{C}))\times\mathbf{A}$$

```
function  [Cn,An]=norm_max(C,A)

coef=1./max(C);      % norm coeff
Cn=C*diag(coef);     % apply to C
if nargin==2
   An=diag(1./coef)*A; % apply inv coeff to A
end
```

```
function  [C,A]=constraints_nonneg(Y,C)

A=nonneg(Y',C')';    % pos spectra (Andersson)
C=nonneg(Y,A);       % pos conc. (Andersson)
```

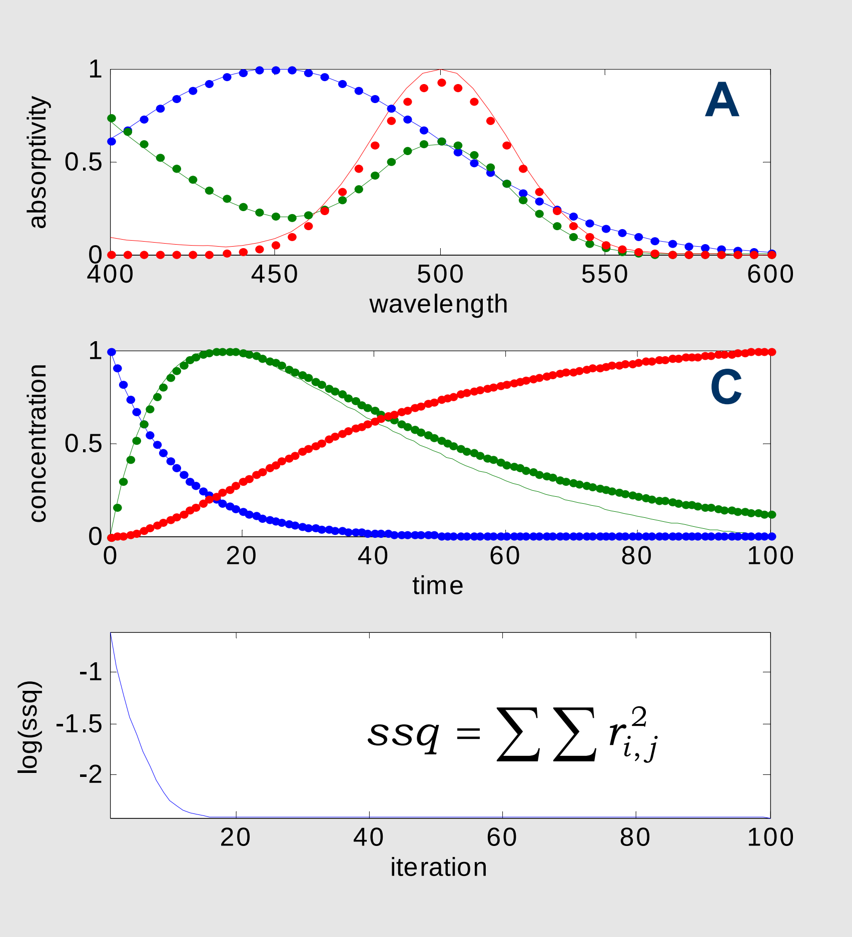
Multivariate Curve Resolution by Alternating Least-Squares (MCR-ALS)

A→*B*→*C*

constraints: **C**, **A** >0, known spectrum of *B*

```
for it=1:100
   C=norm_max(C);           % normalisation

   [C,A]=constraints_known_spec_B(Y,C,A_sim);

   R=Y-C*A;                 % residuals
   ssq(it)=sum(sum(R.*R));
end
```

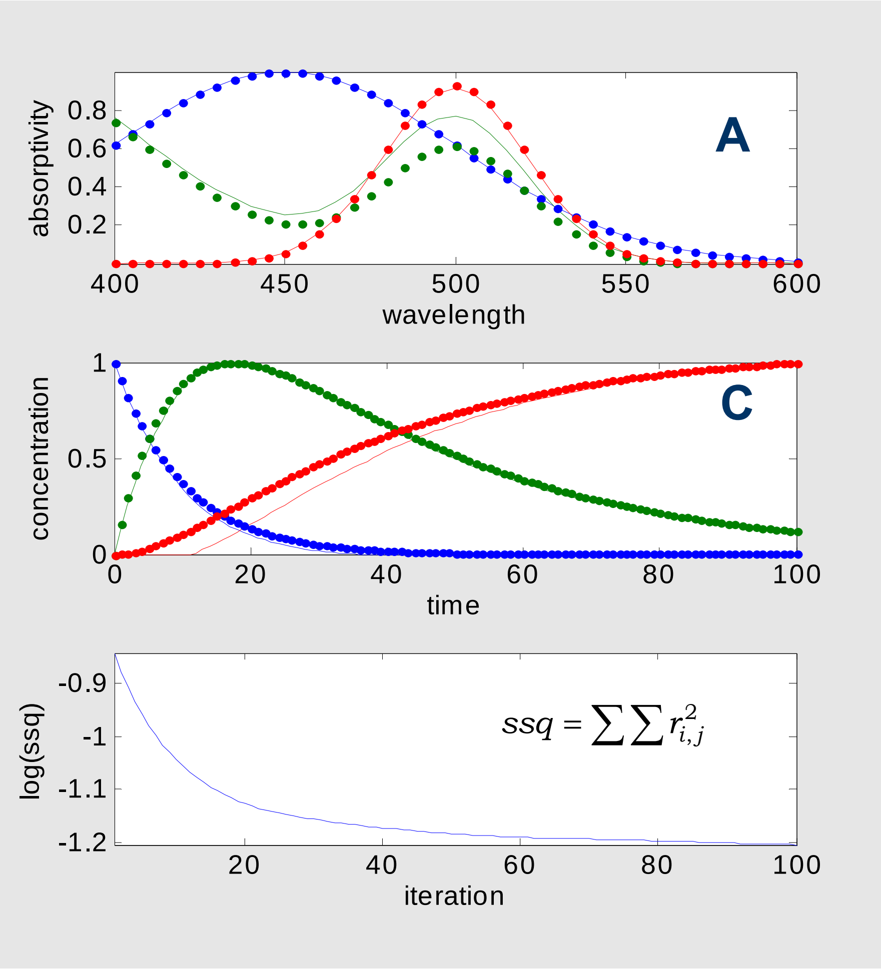
```
function [C,A] =
 constraints_nonneg_known_spec_B(Y,C,A_sim)

A=nonneg(Y',C')';    % pos spectra (Andersson)
A(2,:)=A_sim(2,:);   % known spectrum of B
C=nonneg(Y,A);       % pos conc. (Andersson)
```



Multivariate Curve Resolution by Alternating Least-Squares (MCR-ALS)

A→*B*→*C*

constraints: **C**, **A** >0, known spectrum of *A* & *C*



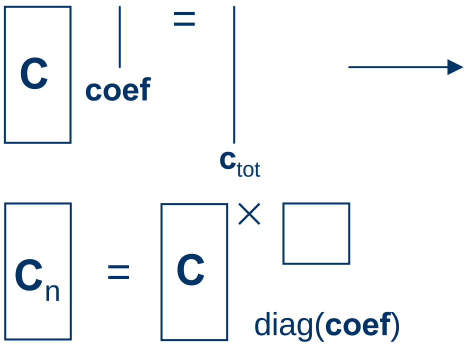
$$ssq = \sum\sum r_{i,j}^2$$

Multivariate Curve Resolution by Alternating Least-Squares (MCR-ALS)

```
for it=1:100
    C=norm_closure(C,[],c_tot); % norm. C to Ctot

    [C,A]=constraints_nonneg_known_spec_B(Y,C,A_sim);

    R=Y-C*A;                    % residuals
    ssq(it)=sum(sum(R.*R));
end
[C_n,A_n]=norm_closure(C,A,c_tot);
```

A→*B*→*C*
constraints: **C**, **A** >0,
known spectrum of *B, closure*

Normalisation to the total conc. (c_{tot}=[*A*]+[*B*]+[*C*]=[*A*]$_0$)

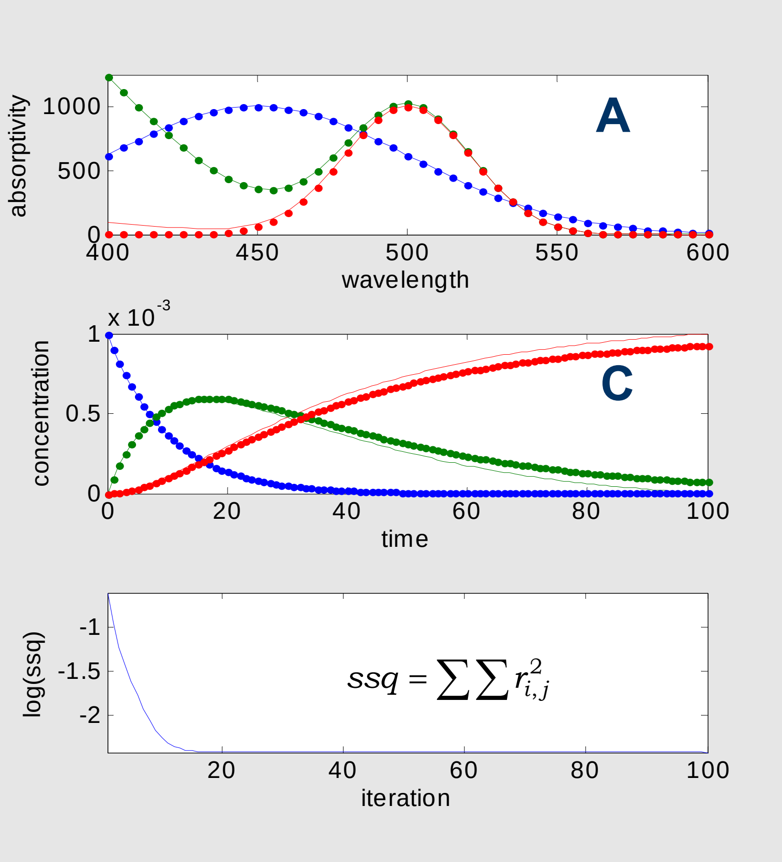


$$\mathbf{coef} = \mathbf{C}^{+} \times \mathbf{c}_{tot}$$

and

$$\mathbf{A}_{\mathrm{n}} = diag(\mathbf{coef})^{-1} \times \mathbf{A}$$

```
function  [Cn,An]=norm_closure(C,A,c_tot)

coef=C\(ones(size(C,1),1)*c_tot); % norm. coeff.
Cn=C*diag(coef);                  % apply to C
if ~isempty(A)
   An=diag(1./coef)*A;            % apply inv. to A
end
```



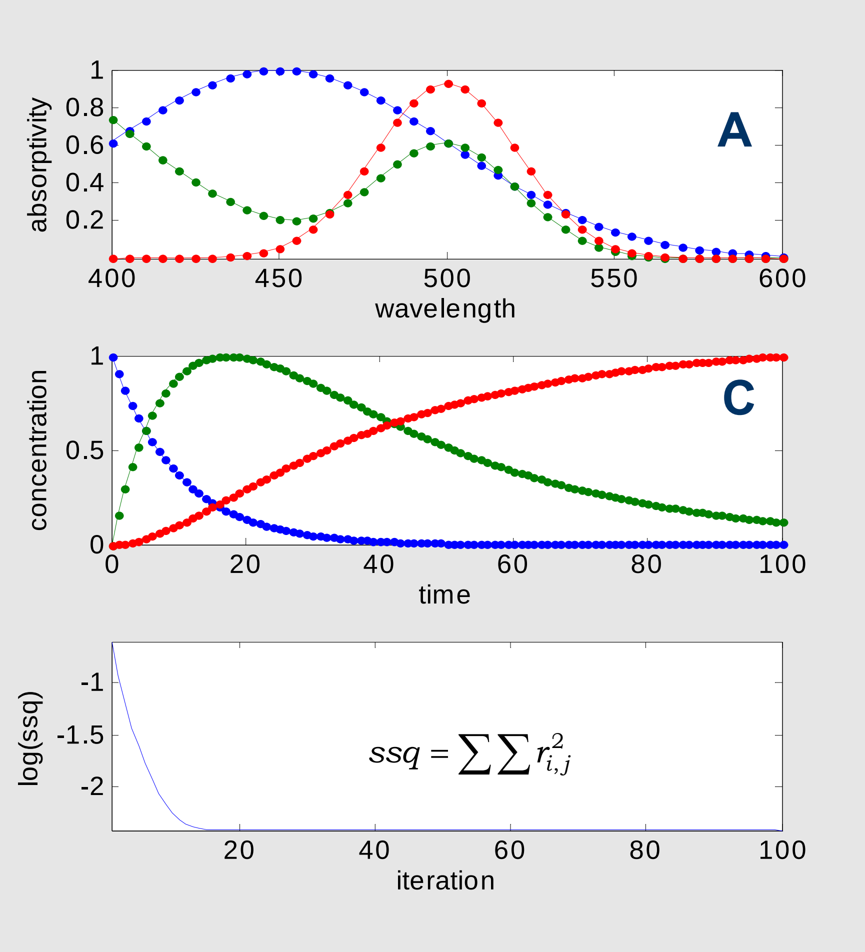
$$ssq = \sum\sum r_{i,j}^{2}$$

Multivariate Curve Resolution by Alternating Least-Squares (MCR-ALS)

A→*B*→*C*

constraints: **C**, **A** >0, known spectrum of *B* & *C*

As full conversion is not reached for intermediate *B* and product *C*, corresponding known spectra are required for full resolution !



Conclusions: ‘Soft’-Modelling

- Advantages
  - No prior knowledge on the chemical system required
  - Estimation of the number of linearly dependent absorbing species and their approximate evolution from PCA, EFA & ALS
  - Info for the development of a ‘hard’ model
  - ‘Better than nothing’
- Drawbacks
  - No physical model
  - No predictions for other exp. conditions possible
  - Uniqueness of the result is rarely given and difficult to validate

Reading Material

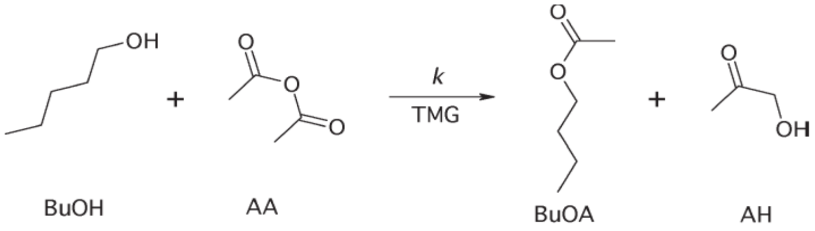
- **Factor Analysis in Chemistry**
  E.R. Malinowski, 3rd ed., Wiley, New York 2002
- **Practical Data Analysis in Chemistry**
  M. Maeder, Y.M. Neuhold, Elsevier, Amsterdam 2007
- **Practical Guide to Chemometrics**
  P. Gemperline (editor), 2nd ed., CRC Press, Boca Raton 2006
- **The Investigation of Organic Reactions and their Mechanisms**
  H. Maskill (editor), Blackwell Publishing, Oxford 2006
- **Evolving factor analysis for the resolution of overlapping chromatographic peaks**
  M. Maeder, Anal. Chem. 59 (1987), 527-530
- **Nonlinear Least-Squares Fitting of Multivariate Absorption Data**
  M. Maeder, A. Zuberbühler. Anal. Chem. 62 (1990), 2220-2224
- **Analyses of 3-way data from equilibrium and kinetic investigations**
  R. Dyson , M. Maeder, Y.M. Neuhold, G. Puxty. Anal. Chim. Act. 490 (2003), 99-108
- **Empirical kinetic modelling of on-line simultaneous infrared and calorimetric measurement using a pareto optimal approach and multi-objective genetic algorithm**
  S.I. Gianoli, G. Puxty, U. Fischer, M. Maeder, K. Hungerbühler. Chemom. Int. Lab. Syst. 85 (2007), 47-62
- **Tutorial on the Fitting of Kinetic Models to Multivariate Spectroscopic Measurements with Non-Linear Regression**
  G. Puxty, M. Maeder, K. Hungerbühler. Chemom. Int. Lab. Syst. 81 (2006), 149-164
- **Data Oriented Process Development: Determination of Reaction Parameters by Small-Scale Calorimetry with in situ Spectroscopy**
  G. Puxty, U. Fischer, M. Jecklin, K. Hungerbühler. Chimia 60 (2006), 605-610

Soft- & Hard Modelling of Kinetic Data

Part 2: Hard Modelling

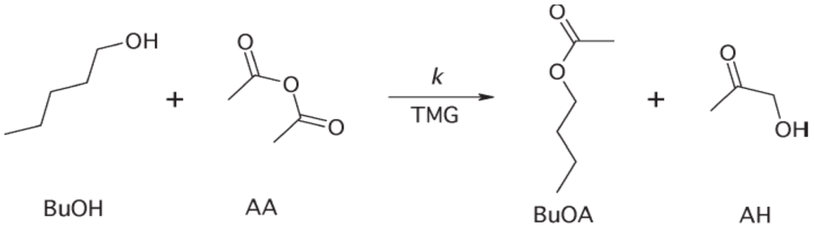
Introduction

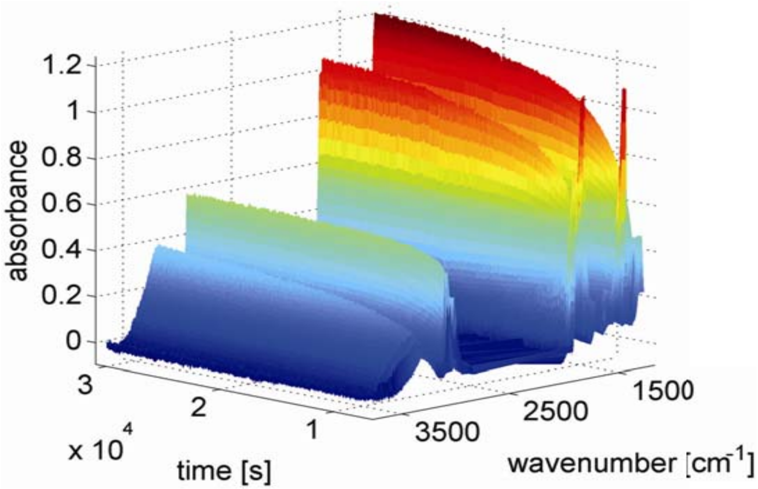
- You follow the reaction between BuOH and the Acetic Acid in the IR range and you get the following absorbance profile :



Introduction

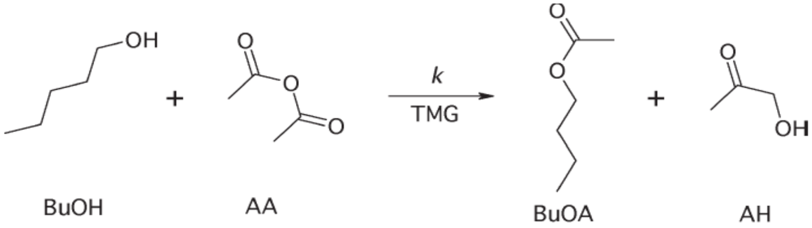
- You follow the reaction between BuOH and the Acetic Acid in the IR range and you get the following absorbance profile :

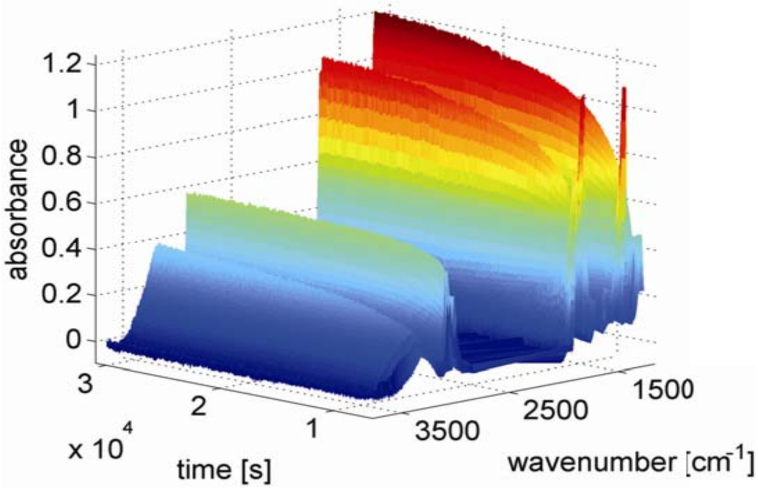




Introduction

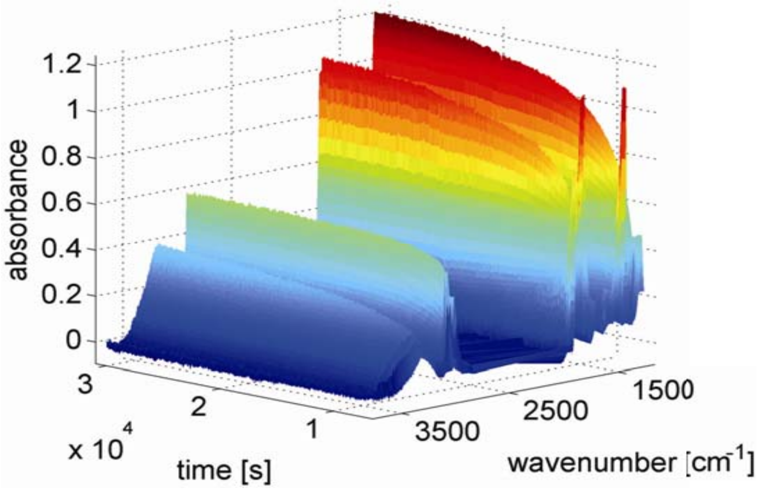
- You follow the reaction between BuOH and the Acetic Acid in the IR range and you get the following absorbance profile :





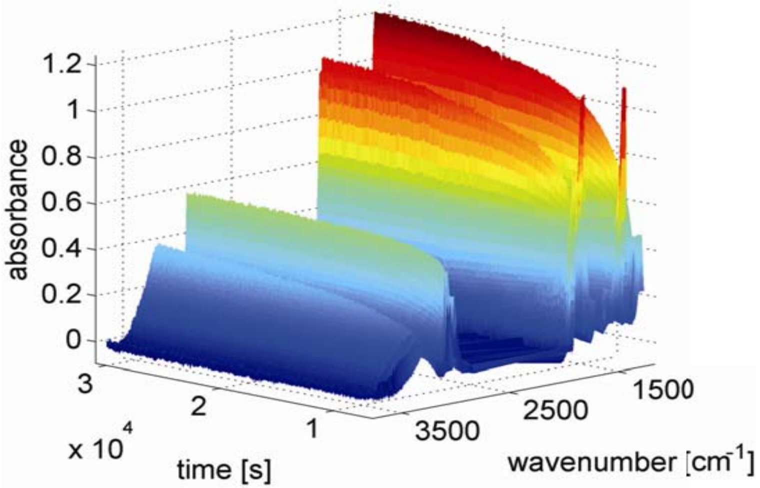
How can the rate constant be extracted from this profile ?

UNIVARIATE approach



UNIVARIATE approach

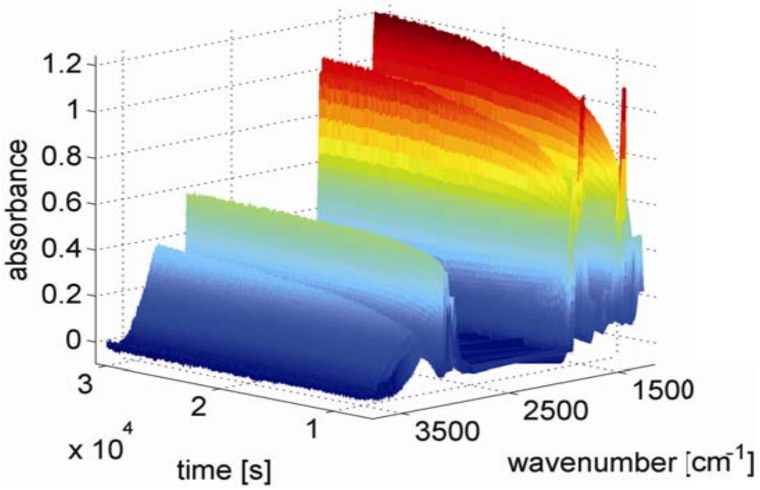
1. You determine the extinction coefficients of all species with known concentrations



UNIVARIATE approach

1. You determine the extinction coefficients of all species with known concentrations

   ⇒ all ε_λ

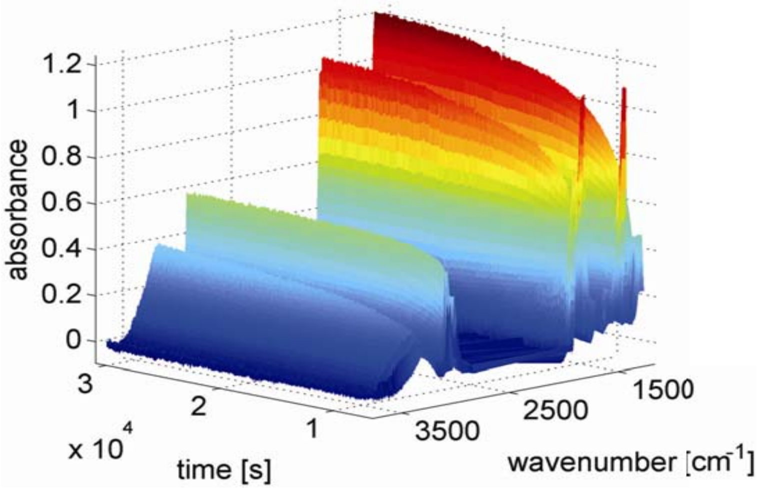


UNIVARIATE approach

1. You determine the extinction coefficients of all species with known concentrations

   $\Rightarrow$ all ε_λ

2. You integrate the set of differential equations (2^{nd} order reaction) with the used initial concentrations



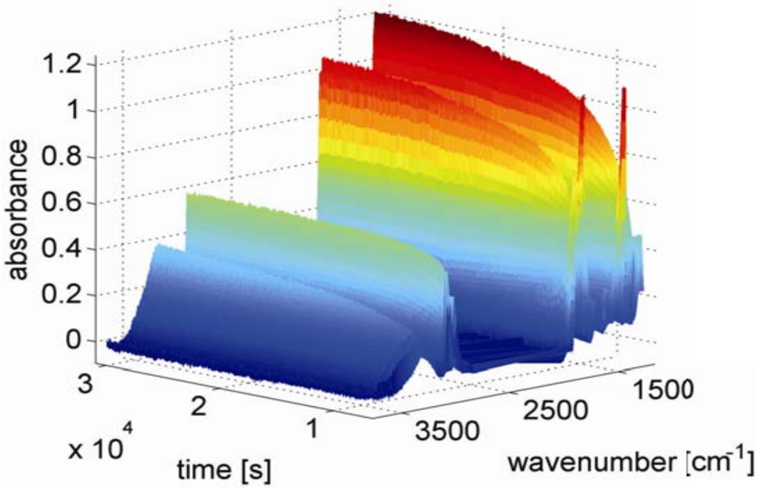
UNIVARIATE approach

1. You determine the extinction coefficients of all species with known concentrations

   ⇒ all ε_λ

2. You integrate the set of differential equations (2^{nd} order reaction) with the used initial concentrations

   ⇒ all $c(t, k)$



UNIVARIATE approach

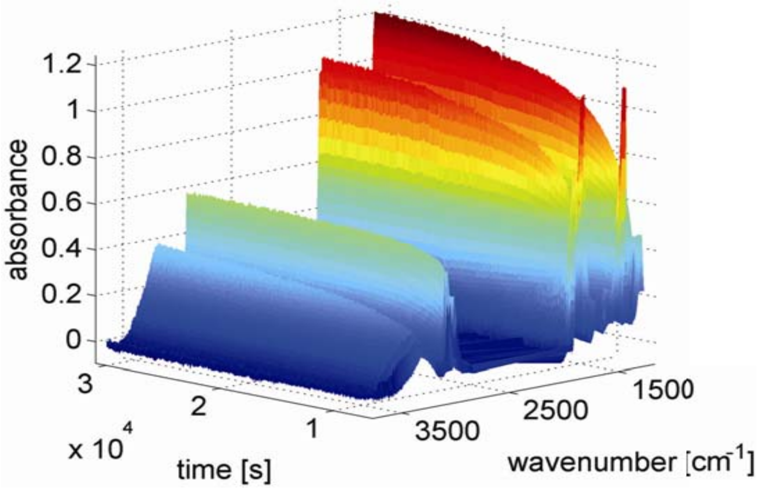
1. You determine the extinction coefficients
   of all species with known concentrations

   ⇒ all ε_λ

2. You integrate the set of differential equations
   (2nd order reaction) with the used initial concentrations

   ⇒ all $c(t, k)$

3. You apply Beer's law at one wavelength



UNIVARIATE approach

1. You determine the extinction coefficients of all species with known concentrations

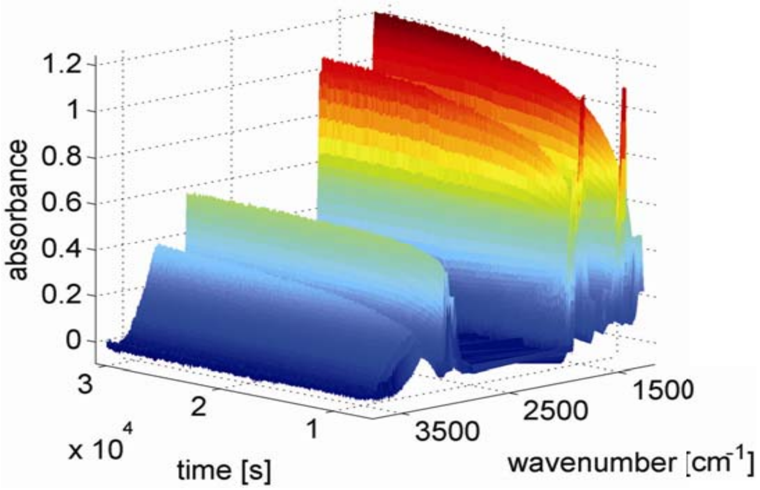
   ⇒ all ε_λ

2. You integrate the set of differential equations (2^{nd} order reaction) with the used initial concentrations

   ⇒ all *c(t, k)*

3. You apply Beer's law at one wavelength

$$abs_\lambda(t) = c_{BuOH}(t,\ k)\cdot\varepsilon_{BuOH,\lambda} + c_{AA,\lambda}(t,\ k)\cdot\varepsilon_{AA,\lambda} + c_{BuOA}(t,\ k)\cdot\varepsilon_{BuOA,\lambda} + c_{HA}(t,\ k)\cdot\varepsilon_{HA,\lambda}$$



UNIVARIATE approach

1. You determine the extinction coefficients of all species with known concentrations

   ⇒ all ε_λ

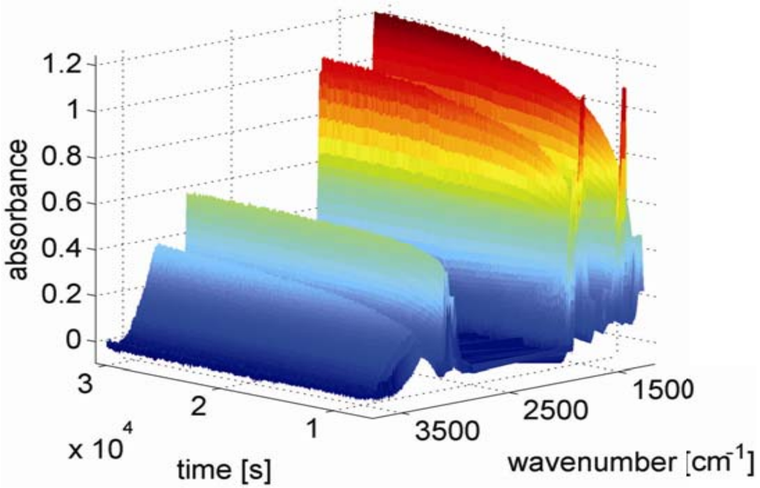
2. You integrate the set of differential equations (2^{nd} order reaction) with the used initial concentrations

   ⇒ all $c(t, k)$

3. You apply Beer's law at one wavelength

$$abs_\lambda(t) = c_{BuOH}(t, k)\cdot\varepsilon_{BuOH,\lambda} + c_{AA,\lambda}(t, k)\cdot\varepsilon_{AA,\lambda} + c_{BuOA}(t, k)\cdot\varepsilon_{BuOA,\lambda} + c_{HA}(t, k)\cdot\varepsilon_{HA,\lambda}$$

4. You find k that best approximate the measured $abs_\lambda(t)$ in the least squares sense



UNIVARIATE approach

1. **CALIBRATION**

   You determine the extinction coefficients of all species with known concentrations

   ⇒ all ε_λ

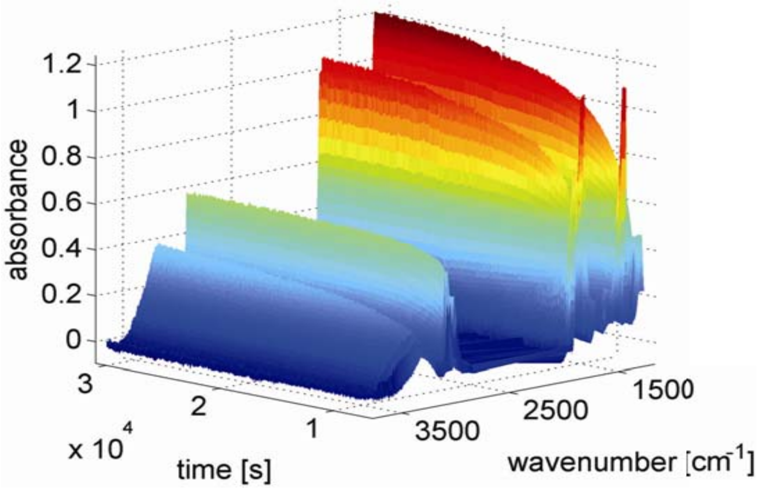
2. You integrate the set of differential equations (2^{nd} order reaction) with the used initial concentrations

   ⇒ all *c(t, k)*

3. You apply Beer's law at one wavelength

$$abs_\lambda(t) = c_{BuOH}(t,\ k)\cdot\varepsilon_{BuOH,\lambda} + c_{AA,\lambda}(t,\ k)\cdot\varepsilon_{AA,\lambda}$$
$$+ c_{BuOA}(t,\ k)\cdot\varepsilon_{BuOA,\lambda} + c_{HA}(t,\ k)\cdot\varepsilon_{HA,\lambda}$$

4. You find *k* that best approximate the measured $abs_\lambda(t)$ in the least squares sense



UNIVARIATE approach

1. **CALIBRATION**

   xtinction coefficients of all species with known concentrations

   $\Rightarrow$ all ε_λ

2. **INTEGRATION OF THE KINETIC MODEL**

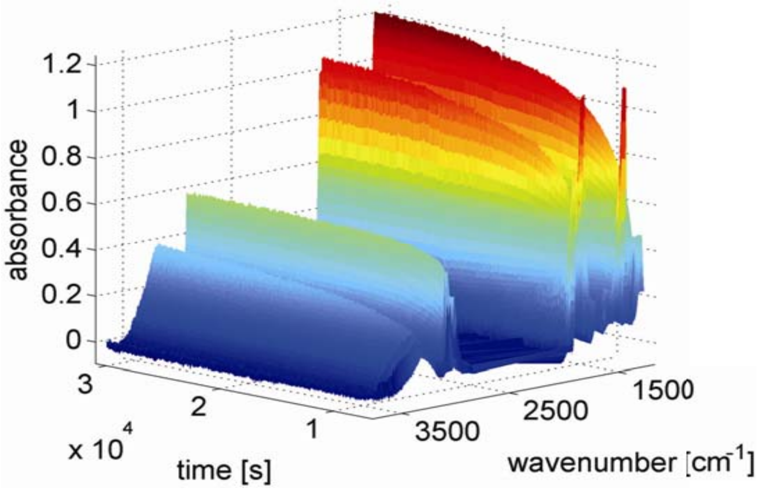
   al equations (2nd order reaction) with the used initial concentrations

   $\Rightarrow$ all $c(t,\ k)$

3. You apply Beer's law at one wavelength

$$abs_\lambda(t) = c_{BuOH}(t,\ k)\cdot\varepsilon_{BuOH,\lambda} + c_{AA,\lambda}(t,\ k)\cdot\varepsilon_{AA,\lambda}$$
$$+ c_{BuOA}(t,\ k)\cdot\varepsilon_{BuOA,\lambda} + c_{HA}(t,\ k)\cdot\varepsilon_{HA,\lambda}$$

4. You find *k* that best approximate the measured *abs*$_\lambda$*(t)* in the least squares sense



UNIVARIATE approach

1. **CALIBRATION** xtinction coefficients
   of all species with known concentrations

   $\Longrightarrow$ all ε_λ

2. **INTEGRATION OF THE KINETIC MODEL** al equations
   the used initial concentrations

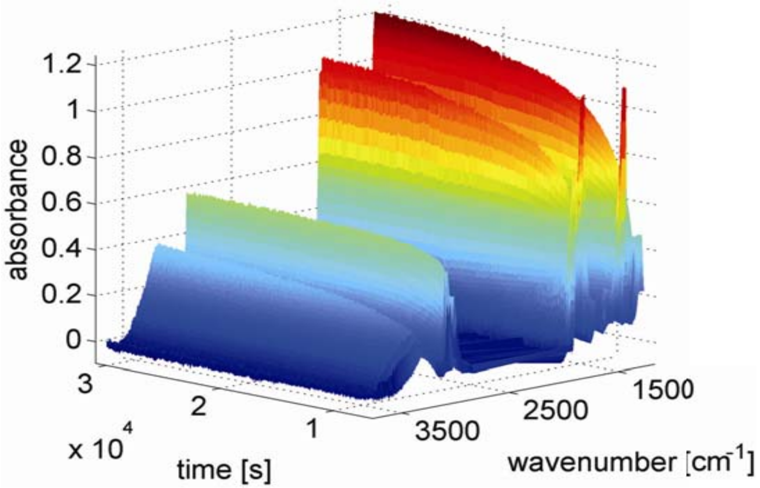
   $\Longrightarrow$ all $c(t, k)$

3. **APPLICATION OF BEER'S LAW IN A UNIVARIATE FORM (= ONE WAVELENGTH)**

   $$+ c_{AA,\lambda}(t, k) \cdot \varepsilon_{AA,\lambda}$$

   $$+ c_{BuOA}(t, k) \cdot \varepsilon_{BuOA,\lambda} + c_{HA}(t, k) \cdot \varepsilon_{HA,\lambda}$$

4. You find *k* that best approximate the measured *abs*$_\lambda$*(t)* in the least squares sense



UNIVARIATE approach

1. **CALIBRATION**

   xtinction coefficients of all species with known concentrations

   ⟹ all ε_λ

2. **INTEGRATION OF THE KINETIC MODEL**

   al equations (2nd order reaction) with the used initial concentrations

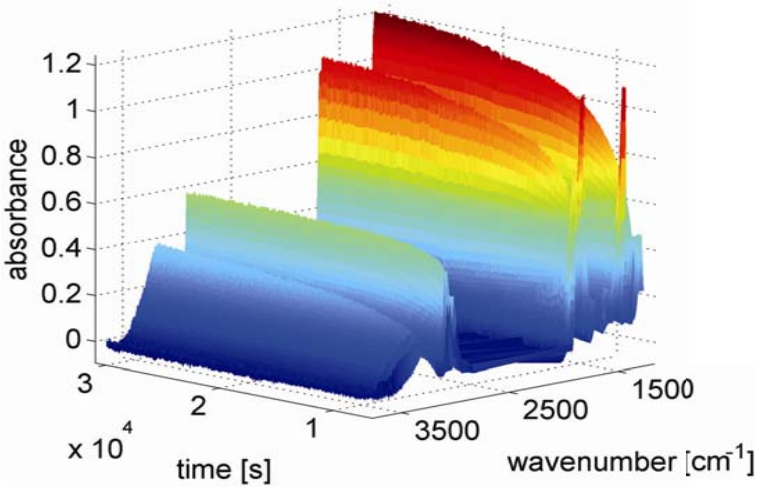
   ⟹ all $c(t, k)$

3. **APPLICATION OF BEER'S LAW IN A UNIVARIATE FORM (= ONE WAVELENGTH)**

   $$+ c_{AA,\lambda}(t,\ k)\cdot\varepsilon_{AA,\lambda} + c_{BuOA}(t,\ k)\cdot\varepsilon_{BuOA,\lambda} + c_{HA}(t,\ k)\cdot\varepsilon_{HA,\lambda}$$

4. **FITTING (= NONLINEAR OPTIMISATION)**

   at best approximate the measured $abs_\lambda(t)$ in the



UNIVARIATE approach

1. **CALIBRATION**

   [illegible] xtinction coefficients of all species with known concentrations

   $\Rightarrow$ all ε_λ

2. **INTEGRATION OF THE KINETIC MODEL**

   [illegible] al equations (2^{nd} order reaction) with the used initial concentrations

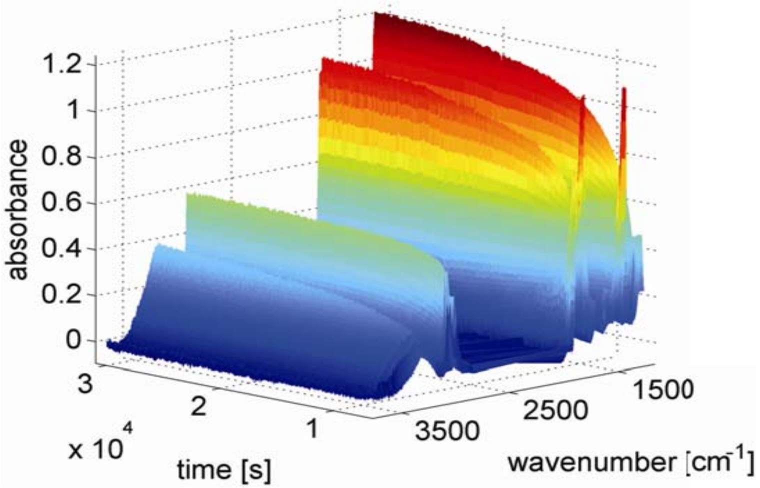
   $\Rightarrow$ all $c(t, k)$

3. **APPLICATION OF BEER'S LAW IN A UNIVARIATE FORM (= ONE WAVELENGTH)**

   $abs_\lambda(t) = c$ [illegible] $+ c_{AA,\lambda}(t, k)\cdot\varepsilon_{AA,\lambda} + c_{BuOA}(t, k)\cdot\varepsilon_{BuOA,\lambda} + c_{HA}(t, k)\cdot\varepsilon_{HA,\lambda}$

4. **FITTING (= NONLINEAR OPTIMISATION)**

   [illegible] at best approximate the measured $abs_\lambda(t)$ in the least [illegible]



KINETIC HARD-MODELING !!!

UNIVARIATE approach

1. **CALIBRATION**

   [illegible] xtinction coefficients of all species with known concentrations

   ⇒ all ε_λ

2. **INTEGRATION OF THE KINETIC MODEL**

   [illegible] ifferential equations (2nd order reaction) with the used initial concentrations

   ⇒ all $c(t, k)$

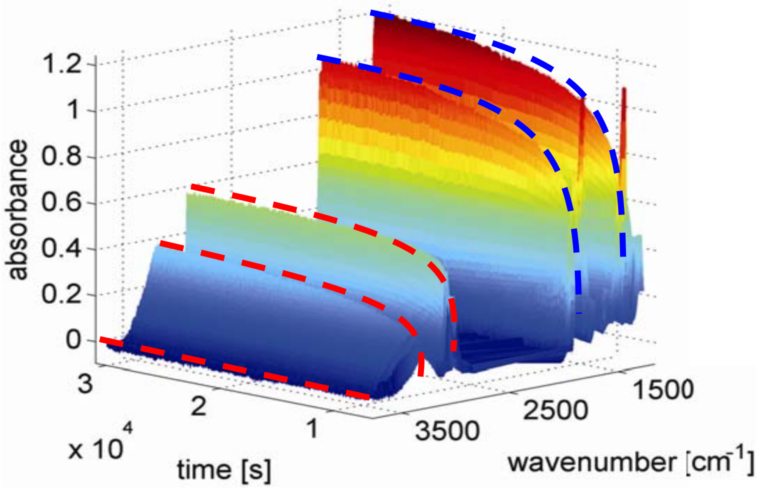
3. **APPLICATION OF BEER'S LAW IN A UNIVARIATE FORM (= ONE WAVELENGTH)**

   [illegible]

   $$abs_\lambda(t) = c_{[illegible]}(t, k)\cdot\varepsilon_{[illegible]} + c_{AA,\lambda}(t, k)\cdot\varepsilon_{AA,\lambda} + c_{BuOA}(t, k)\cdot\varepsilon_{BuOA,\lambda} + c_{HA}(t, k)\cdot\varepsilon_{HA,\lambda}$$

4. **FITTING (= NONLINEAR OPTIMISATION)**

   [illegible] at best approximate the measured $abs_\lambda(t)$ in the [illegible]



KINETIC HARD-MODELING !!!

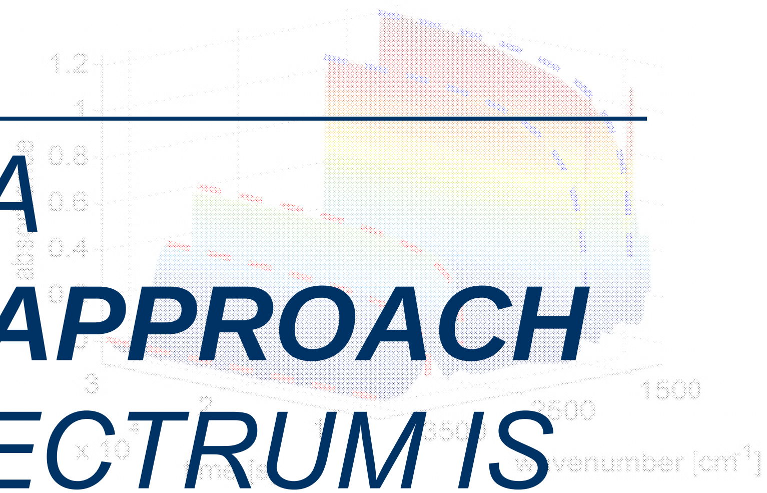
Question :

Which wavelength do you follow ?

What about the rest of the spectrum ?

UNIVARIATE approach

1. CALIBRATION
2. INTEGRATION OF THE KINETIC MODEL
3. APPLICATION OF BEER'S LAW IN A UNIVARIATE WAY (= ONE WAVELENGTH)
4. FITTING (= NONLINEAR OPTIMISATION)



WITH A **MULTIVARIATE APPROACH** THE WHOLE SPECTRUM IS USED TO FIT k

KINETIC HARD-MODELING !!!

Question :

Which wavelength do you follow ?

What about the rest of the spectrum ?

Presentation of a typical kinetic-modeling algorithm

Presentation of a typical kinetic-modeling algorithm

The following topics will be addressed :

Presentation of a typical kinetic-modeling algorithm

The following topics will be addressed :

- Typical fitted signals
  - Multivariate Spectroscopy (Beer’s law)
  - Univariate Calorimetry

Presentation of a typical kinetic-modeling algorithm

The following topics will be addressed :

- Typical fitted signals
  - Multivariate Spectroscopy (Beer’s law)
  - Univariate Calorimetry
- Generalisation of the Rate Law

Presentation of a typical kinetic-modeling algorithm

The following topics will be addressed :

- Typical fitted signals
  - Multivariate Spectroscopy (Beer’s law)
  - Univariate Calorimetry
- Generalisation of the Rate Law
- Numerical integration

Presentation of a typical kinetic-modeling algorithm

The following topics will be addressed :

- Typical fitted signals
  - Multivariate Spectroscopy (Beer’s law)
  - Univariate Calorimetry
- Generalisation of the Rate Law
- Numerical integration
- Separation of parameters

Presentation of a typical kinetic-modeling algorithm

The following topics will be addressed :

- Typical fitted signals
  - Multivariate Spectroscopy (Beer's law)
  - Univariate Calorimetry
- Generalisation of the Rate Law
- Numerical integration
- Separation of parameters
- Newton-Gauss method

Presentation of a typical kinetic-modeling algorithm

The following topics will be addressed :

- Typical fitted signals
  - Multivariate Spectroscopy (Beer's law)
  - Univariate Calorimetry
- Generalisation of the Rate Law
- Numerical integration
- Separation of parameters
- Newton-Gauss method

Two common problems will also be treated :

Presentation of a typical kinetic-modeling algorithm

The following topics will be addressed :

- Typical fitted signals
  - Multivariate Spectroscopy (Beer’s law)
  - Univariate Calorimetry
- Generalisation of the Rate Law
- Numerical integration
- Separation of parameters
- Newton-Gauss method

Two common problems will also be treated :

1. Divergence problems *(Levenberg-Marquardt modification)*

Presentation of a typical kinetic-modeling algorithm

The following topics will be addressed :

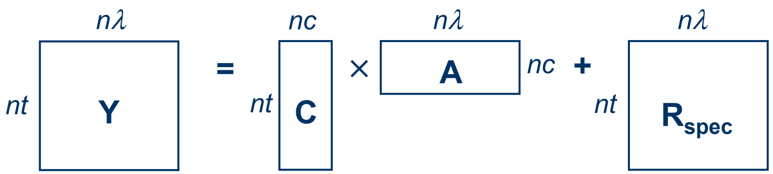
- Typical fitted signals
  - Multivariate Spectroscopy (Beer's law)
  - Univariate Calorimetry
- Generalisation of the Rate Law
- Numerical integration
- Separation of parameters
- Newton-Gauss method

Two common problems will also be treated :

1. Divergence problems *(Levenberg-Marquardt modification)*
2. Rank deficiency *(methods of annihilation)*

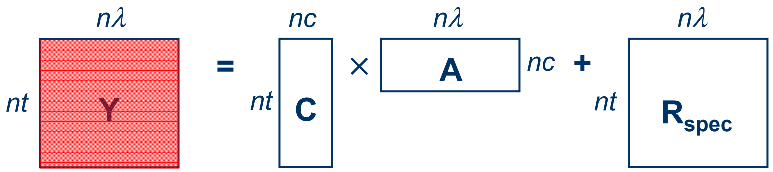
SPECTROSCOPY and CALORIMETRY

- Spectroscopy (IR, UV, Raman, fluorescence …)

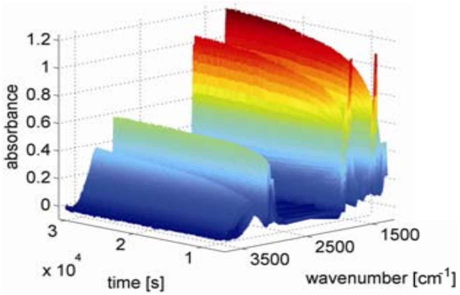


SPECTROSCOPY and CALORIMETRY

- Spectroscopy (IR, UV, Raman, fluorescence …)

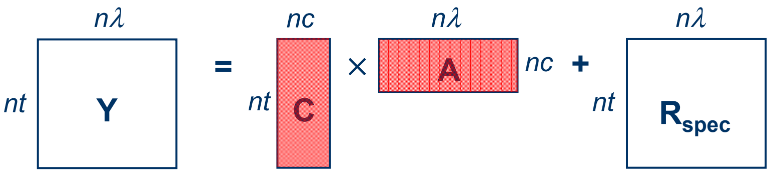


Measurement (Multivariate)



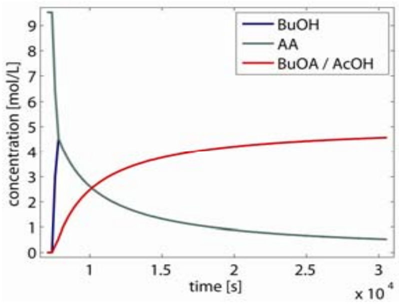
SPECTROSCOPY and CALORIMETRY

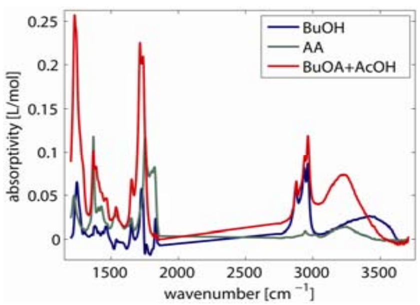
- Spectroscopy (IR, UV, Raman, fluorescence …)



$$\underset{nt \times n\lambda}{Y} = \underset{nt \times nc}{C} \times \underset{nc \times n\lambda}{A} + \underset{nt \times n\lambda}{R_{spec}}$$

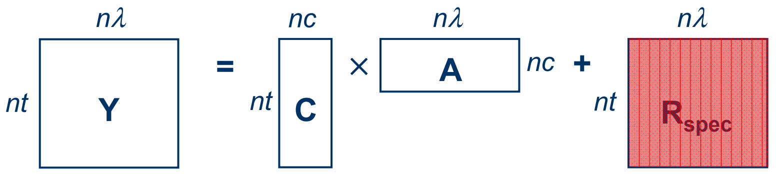
Physical model
(Beer’s law)



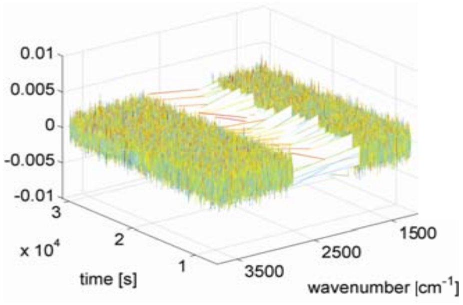


SPECTROSCOPY and CALORIMETRY

- Spectroscopy (IR, UV, Raman, fluorescence …)

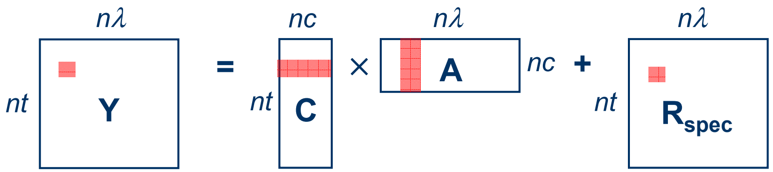


Unmodelled effects (Matrix)



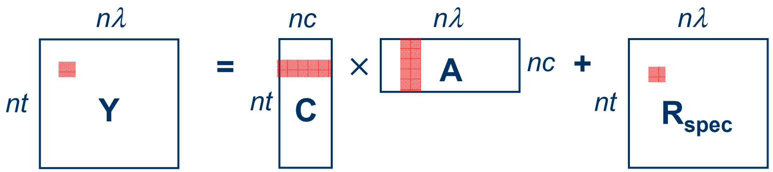
SPECTROSCOPY and CALORIMETRY

- Spectroscopy (IR, UV, Raman, fluorescence …)



SPECTROSCOPY and CALORIMETRY

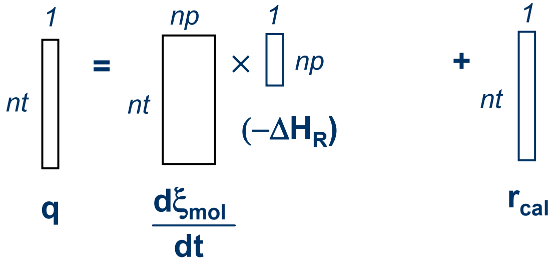
- Spectroscopy (IR, UV, Raman, fluorescence …)



$n\lambda$ nc $n\lambda$ $n\lambda$

nt $\mathbf{Y}$ = nt $\mathbf{C}$ × $\mathbf{A}$ nc + nt $\mathbf{R}_{\mathbf{spec}}$

- Calorimetry



1 np 1 1

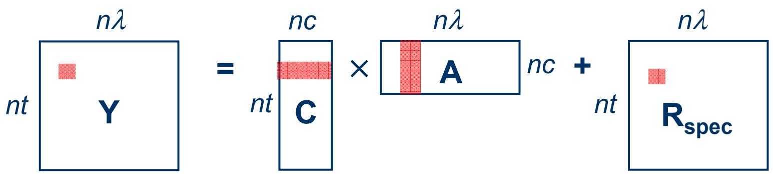
nt = nt × np + nt

$(-\Delta \mathbf{H_R})$

$\mathbf{q}$ $\dfrac{\mathbf{d\xi_{mol}}}{\mathbf{dt}}$ $\mathbf{r_{cal}}$

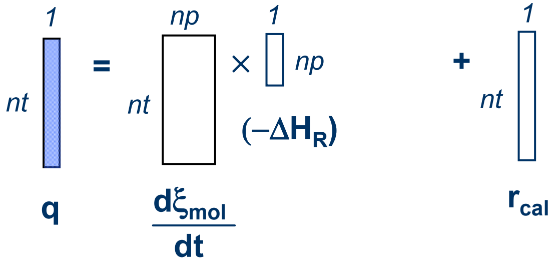
SPECTROSCOPY and CALORIMETRY

- Spectroscopy (IR, UV, Raman, fluorescence …)



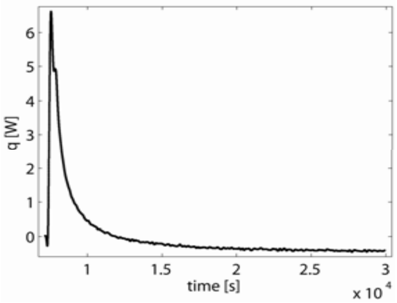
$$\mathbf{Y} = \mathbf{C} \times \mathbf{A} + \mathbf{R}_{spec}$$

- Calorimetry



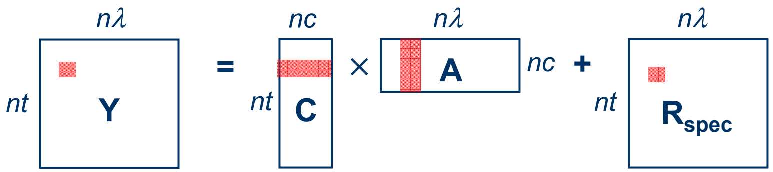
$$\mathbf{q} = \frac{d\xi_{mol}}{dt} \times (-\Delta H_R) + \mathbf{r}_{cal}$$

Measurement (Univariate)

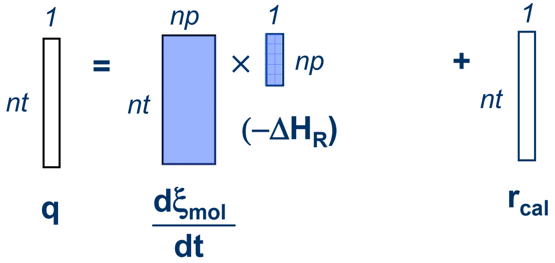


SPECTROSCOPY and CALORIMETRY

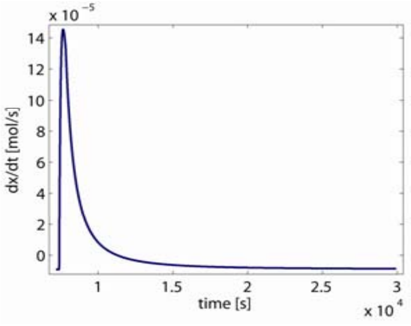
- Spectroscopy (IR, UV, Raman, fluorescence …)



- Calorimetry



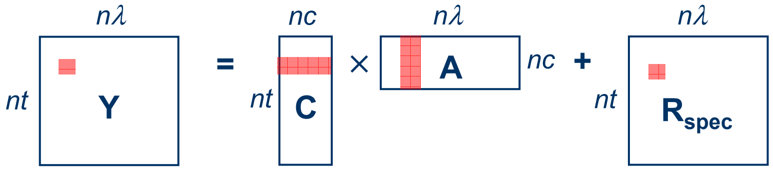
Physical model
(Reaction heat balance)



× 43 kJ/mol

SPECTROSCOPY and CALORIMETRY

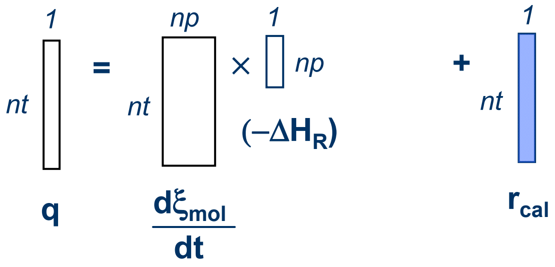
- Spectroscopy (IR, UV, Raman, fluorescence …)



$n\lambda$ nc $n\lambda$ $n\lambda$

nt **Y** = nt **C** × **A** nc + nt $\mathbf{R}_{\mathbf{spec}}$

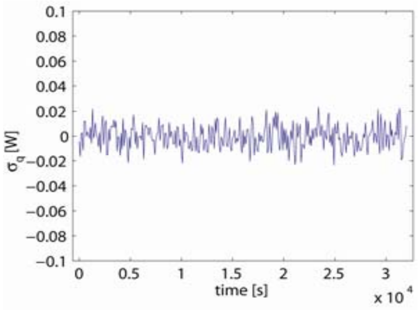
- Calorimetry



1 np 1 1

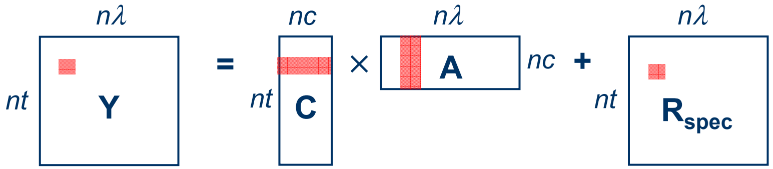
nt **q** = nt $\dfrac{d\xi_{mol}}{dt}$ × np $(-\Delta H_R)$ + nt $\mathbf{r}_{\mathbf{cal}}$

Unmodelled effects
(Vector)

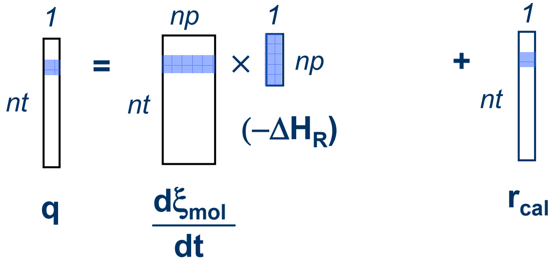


SPECTROSCOPY and CALORIMETRY

- Spectroscopy (IR, UV, Raman, fluorescence …)



- Calorimetry



$$\frac{\mathrm{d}\xi_{\mathrm{mol}}}{\mathrm{dt}} = \mathbf{V}(t)\cdot\mathbf{r}(t)$$

Definition of the optimisation problem

- The residuals are defined as the difference between the measurement and the model

$$\mathbf{Y} = \mathbf{C} \cdot \mathbf{A} + \mathbf{R}_{\mathbf{spec}}$$

$$\mathbf{q} = \frac{\mathbf{d}\xi_{\mathbf{mol}}}{\mathbf{dt}} \cdot \left(-\Delta \mathbf{H}_{\mathbf{R}}\right) + \mathbf{r}_{\mathbf{cal}}$$

Definition of the optimisation problem

- The residuals are defined as the difference between the measurement and the model

$$\mathbf{R_{spec}} = \mathbf{Y} - \mathbf{C} \cdot \mathbf{A}$$

$$\mathbf{r_{cal}} = \mathbf{q} - \frac{\mathbf{d\xi_{mol}}}{\mathbf{dt}} \cdot \left(-\Delta \mathbf{H_R}\right)$$

Definition of the optimisation problem

- The residuals are defined as the difference between the measurement and the model

$$\mathbf{R}_{\mathbf{spec}} = \mathbf{Y} - \mathbf{C} \cdot \mathbf{A}$$

$$\mathbf{r}_{\mathbf{cal}} = \mathbf{q} - \frac{\mathbf{d\xi_{mol}}}{\mathbf{dt}} \cdot \left(-\mathbf{\Delta H}_{\mathbf{R}}\right)$$

- As the mass balance part is :

$$\mathbf{C} = f\left(model,\ \mathbf{p}\right)$$

$$\frac{\mathbf{d\xi_{mol}}}{\mathbf{dt}} = g\left(model,\ \mathbf{p}\right)$$

Definition of the optimisation problem

- The residuals are defined as the difference between the measurement and the model

$$\mathbf{R}_{\mathbf{spec}} = \mathbf{Y} - \mathbf{C} \cdot \mathbf{A}$$

$$\mathbf{r}_{\mathbf{cal}} = \mathbf{q} - \frac{\mathbf{d\xi_{mol}}}{\mathbf{dt}} \cdot \left(-\mathbf{\Delta H_R}\right)$$

- As the mass balance part is :

$$\mathbf{C} = f\left(model,\ \mathbf{p}\right)$$

$$\frac{\mathbf{d\xi_{mol}}}{\mathbf{dt}} = g\left(model,\ \mathbf{p}\right)$$

- The optimisation problem is :

minimize $\mathbf{R}_{\mathbf{spec}} = f\left(\mathbf{Y}, model,\ \mathbf{p}\right)$

in the least square sense

by changing **p**

minimize $\mathbf{r}_{\mathbf{cal}} = f\left(\mathbf{q}, model,\ \mathbf{p}\right)$

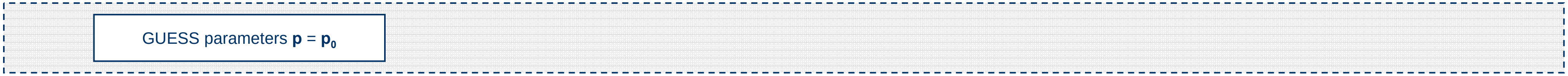
in the least square sense

by changing **p**

Kinetic modeling algorithm

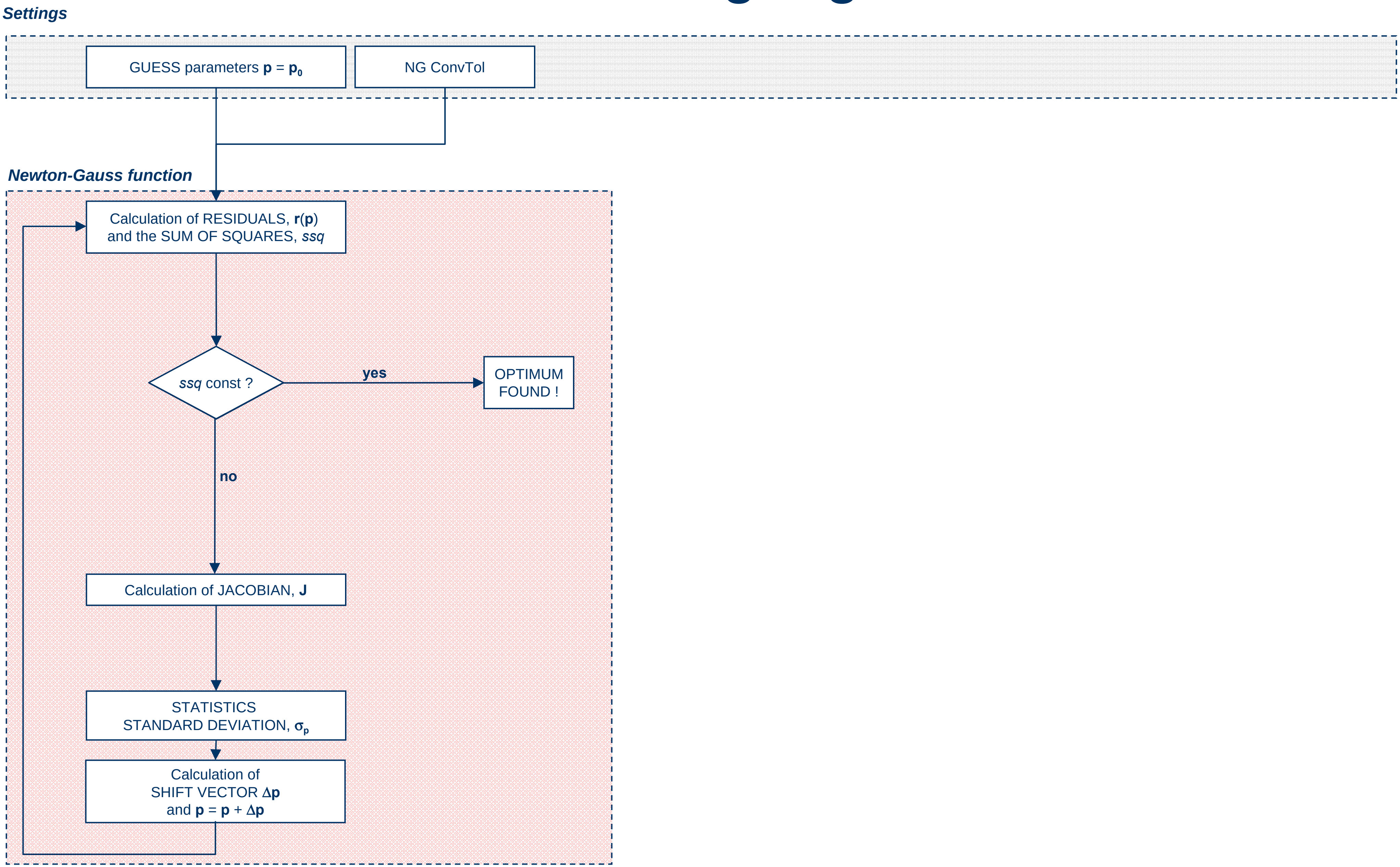
Kinetic modeling algorithm

Settings



GUESS parameters $\mathbf{p} = \mathbf{p}_0$

Kinetic modeling algorithm



Settings

GUESS parameters $\mathbf{p} = \mathbf{p}_0$

NG ConvTol

Newton-Gauss function

Calculation of RESIDUALS, $\mathbf{r}(\mathbf{p})$
and the SUM OF SQUARES, *ssq*

ssq const ?

yes

OPTIMUM FOUND !

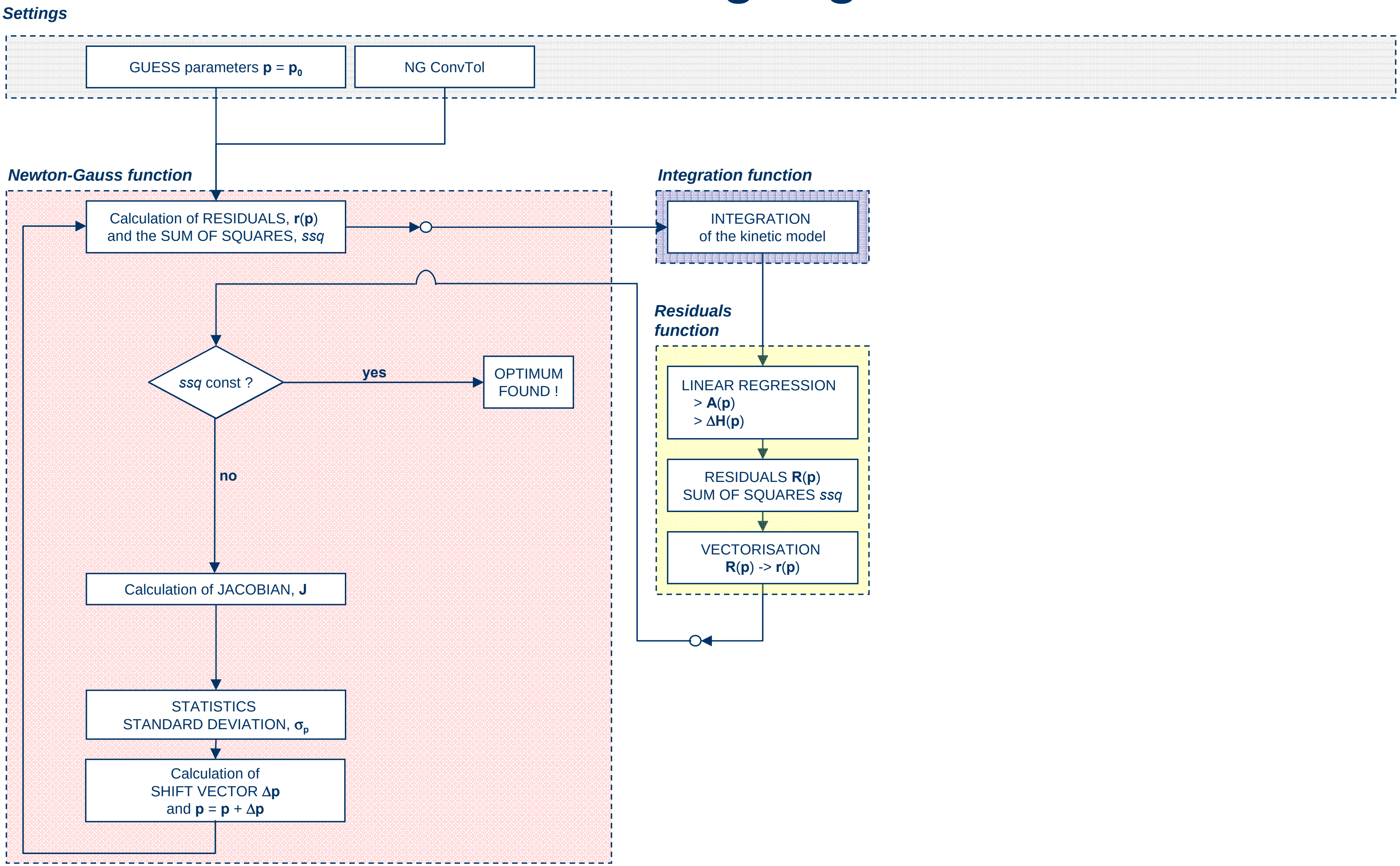
no

Calculation of JACOBIAN, **J**

STATISTICS
STANDARD DEVIATION, σ_p

Calculation of
SHIFT VECTOR $\Delta\mathbf{p}$
and $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$

Kinetic modeling algorithm



Settings

GUESS parameters **p** = $\mathbf{p}_0$

NG ConvTol

Newton-Gauss function

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq const ?

yes

OPTIMUM FOUND !

no

Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

Integration function

INTEGRATION of the kinetic model

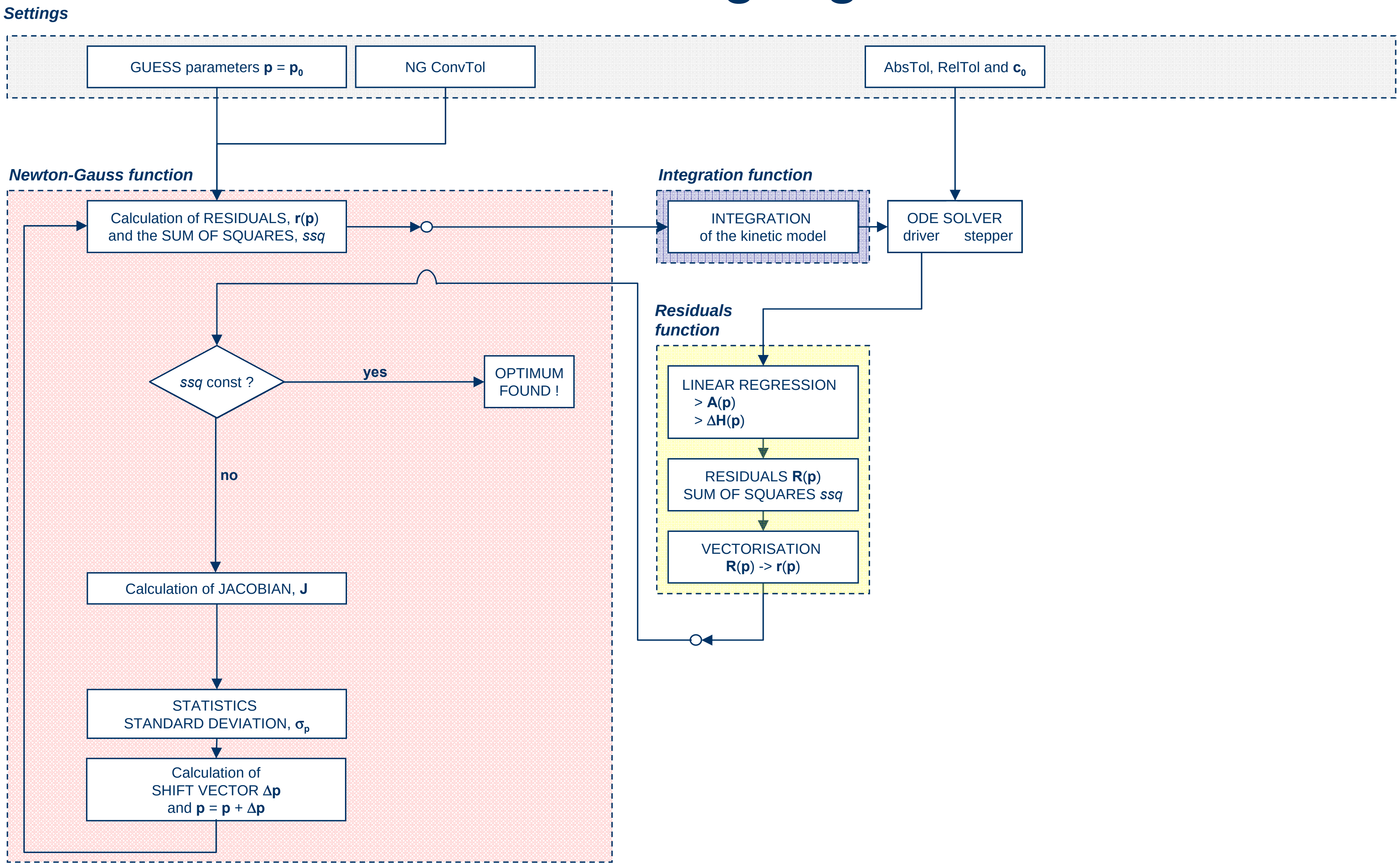
Residuals function

LINEAR REGRESSION
> **A**(**p**)
> Δ**H**(**p**)

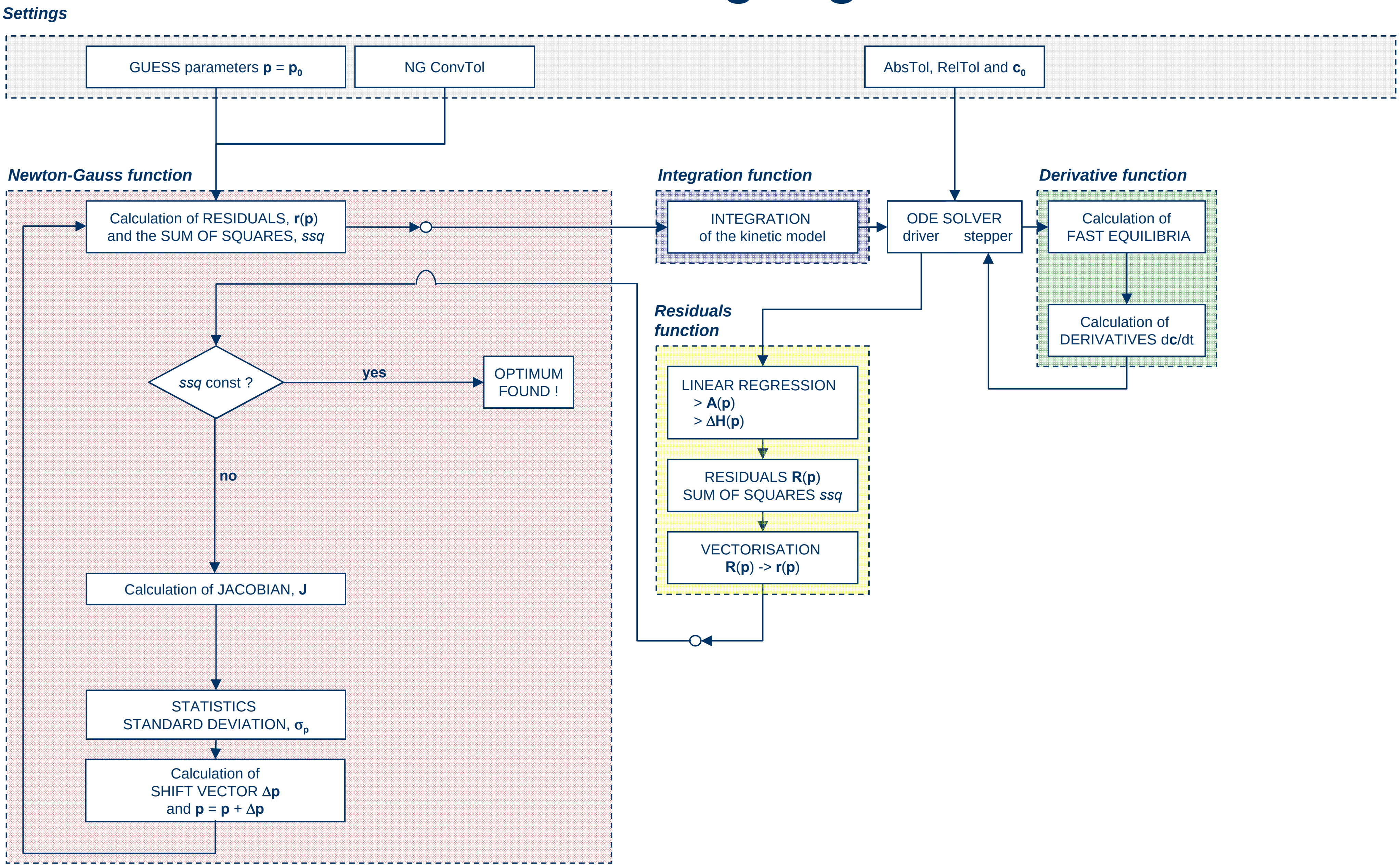
RESIDUALS **R**(**p**) SUM OF SQUARES *ssq*

VECTORISATION **R**(**p**) -> **r**(**p**)

Kinetic modeling algorithm

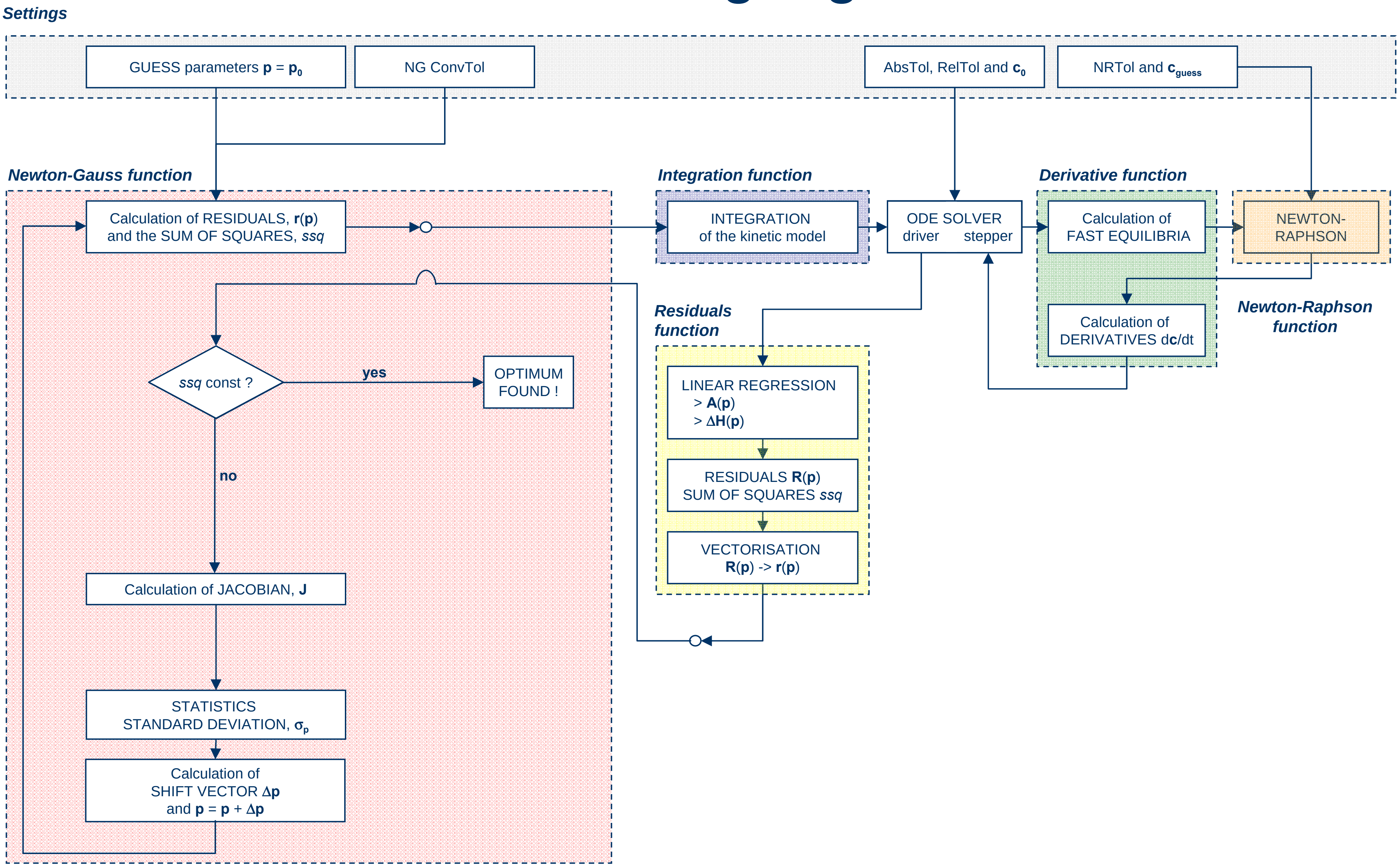

Settings
GUESS parameters $\mathbf{p} = \mathbf{p}_0$
NG ConvTol
AbsTol, RelTol and $\mathbf{c}_0$
Newton-Gauss function
Calculation of RESIDUALS, $\mathbf{r}(\mathbf{p})$
and the SUM OF SQUARES, *ssq*
ssq const ?
yes
OPTIMUM
FOUND !
no
Calculation of JACOBIAN, $\mathbf{J}$
STATISTICS
STANDARD DEVIATION, σ_p
Calculation of
SHIFT VECTOR $\Delta\mathbf{p}$
and $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$
Integration function
INTEGRATION
of the kinetic model
ODE SOLVER
driver stepper
Residuals function
LINEAR REGRESSION
> $\mathbf{A}(\mathbf{p})$
> $\Delta\mathbf{H}(\mathbf{p})$
RESIDUALS $\mathbf{R}(\mathbf{p})$
SUM OF SQUARES *ssq*
VECTORISATION
$\mathbf{R}(\mathbf{p})$ -> $\mathbf{r}(\mathbf{p})$

Kinetic modeling algorithm



Settings
GUESS parameters **p** = **p**$_0$
NG ConvTol
AbsTol, RelTol and **c**$_0$
Newton-Gauss function
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
ssq const ?
yes
OPTIMUM FOUND !
no
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**
Integration function
INTEGRATION of the kinetic model
ODE SOLVER driver stepper
Derivative function
Calculation of FAST EQUILIBRIA
Calculation of DERIVATIVES d**c**/dt
Residuals function
LINEAR REGRESSION > **A**(**p**) > Δ**H**(**p**)
RESIDUALS **R**(**p**) SUM OF SQUARES *ssq*
VECTORISATION **R**(**p**) -> **r**(**p**)

Kinetic modeling algorithm



Settings

GUESS parameters $\mathbf{p} = \mathbf{p}_0$

NG ConvTol

AbsTol, RelTol and $\mathbf{c}_0$

NRTol and $\mathbf{c}_{\mathbf{guess}}$

Newton-Gauss function

Calculation of RESIDUALS, $\mathbf{r}(\mathbf{p})$ and the SUM OF SQUARES, *ssq*

ssq const ?

yes

OPTIMUM FOUND !

no

Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR $\Delta\mathbf{p}$ and $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$

Integration function

INTEGRATION of the kinetic model

ODE SOLVER driver stepper

Derivative function

Calculation of FAST EQUILIBRIA

Calculation of DERIVATIVES d**c**/dt

NEWTON-RAPHSON

Newton-Raphson function

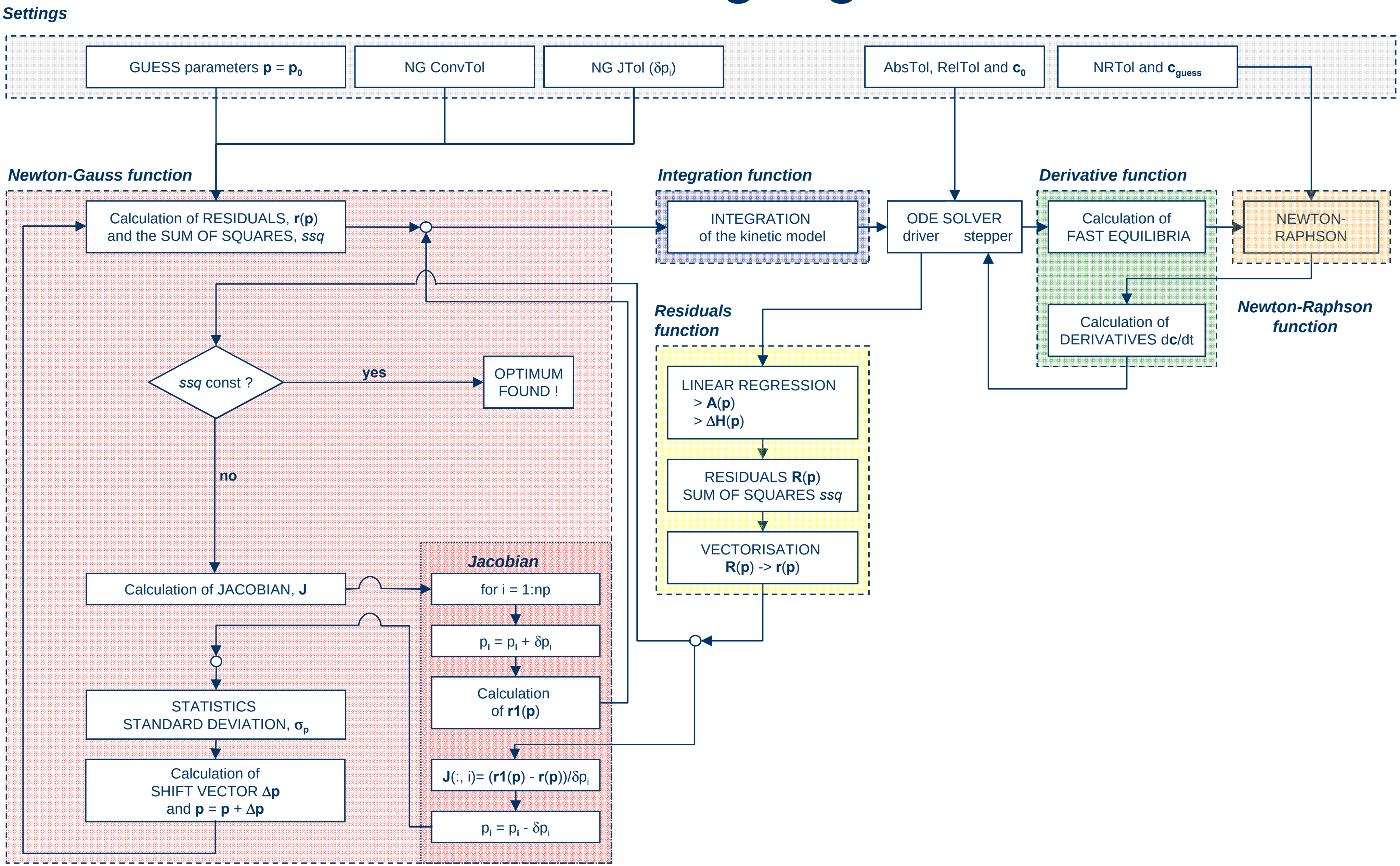
Residuals function

LINEAR REGRESSION
> $\mathbf{A}(\mathbf{p})$
> $\Delta\mathbf{H}(\mathbf{p})$

RESIDUALS $\mathbf{R}(\mathbf{p})$ SUM OF SQUARES *ssq*

VECTORISATION $\mathbf{R}(\mathbf{p})$ -> $\mathbf{r}(\mathbf{p})$

Kinetic modeling algorithm



Settings

GUESS parameters **p** = $\mathbf{p}_0$

NG ConvTol

NG JTol (δp_i)

AbsTol, RelTol and $\mathbf{c}_0$

NRTol and $\mathbf{c}_{guess}$

Newton-Gauss function

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq const ?

yes

OPTIMUM FOUND !

no

Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR $\Delta\mathbf{p}$ and $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$

Jacobian

for i = 1:np

$p_i = p_i + \delta p_i$

Calculation of **r1**(**p**)

$\mathbf{J}(:, i) = (\mathbf{r1}(\mathbf{p}) - \mathbf{r}(\mathbf{p}))/\delta p_i$

$p_i = p_i - \delta p_i$

Integration function

INTEGRATION of the kinetic model

ODE SOLVER driver stepper

Derivative function

Calculation of FAST EQUILIBRIA

Calculation of DERIVATIVES d**c**/dt

NEWTON-RAPHSON

Newton-Raphson function

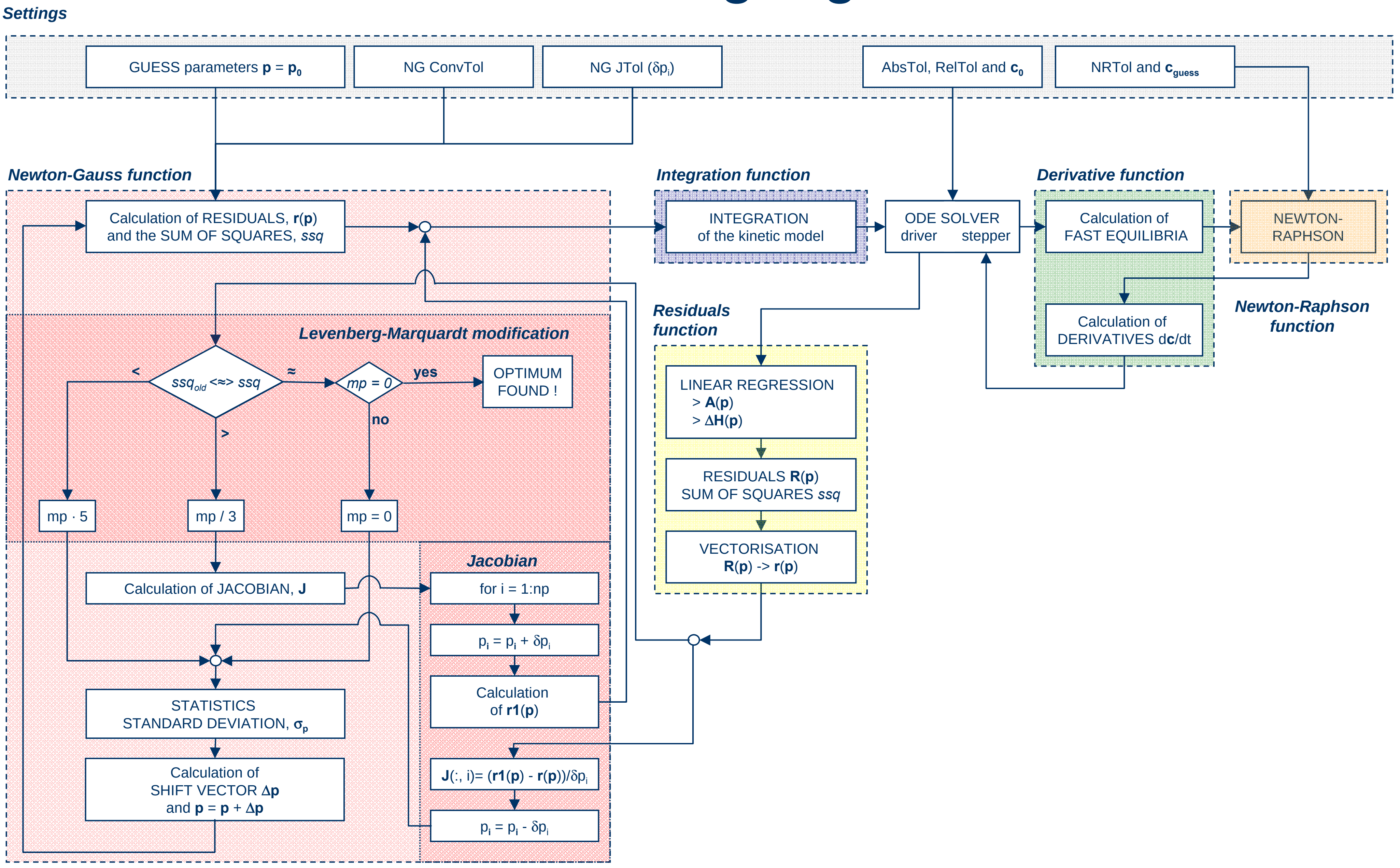
Residuals function

LINEAR REGRESSION
> **A**(**p**)
> Δ**H**(**p**)

RESIDUALS **R**(**p**) SUM OF SQUARES *ssq*

VECTORISATION **R**(**p**) -> **r**(**p**)

Kinetic modeling algorithm



Settings

GUESS parameters $\mathbf{p} = \mathbf{p}_0$

NG ConvTol

NG JTol (δp_i)

AbsTol, RelTol and $\mathbf{c}_0$

NRTol and $\mathbf{c}_{guess}$

Newton-Gauss function

Calculation of RESIDUALS, $\mathbf{r}(\mathbf{p})$ and the SUM OF SQUARES, *ssq*

Levenberg-Marquardt modification

ssq_{old} <≈> ssq

<

≈

>

$mp = 0$

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR $\Delta\mathbf{p}$ and $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$

Jacobian

for i = 1:np

$p_i = p_i + \delta p_i$

Calculation of $\mathbf{r1}(\mathbf{p})$

$\mathbf{J}(:, i) = (\mathbf{r1}(\mathbf{p}) - \mathbf{r}(\mathbf{p}))/\delta p_i$

$p_i = p_i - \delta p_i$

Integration function

INTEGRATION of the kinetic model

ODE SOLVER driver stepper

Derivative function

Calculation of FAST EQUILIBRIA

Calculation of DERIVATIVES $d\mathbf{c}/dt$

NEWTON-RAPHSON

Newton-Raphson function

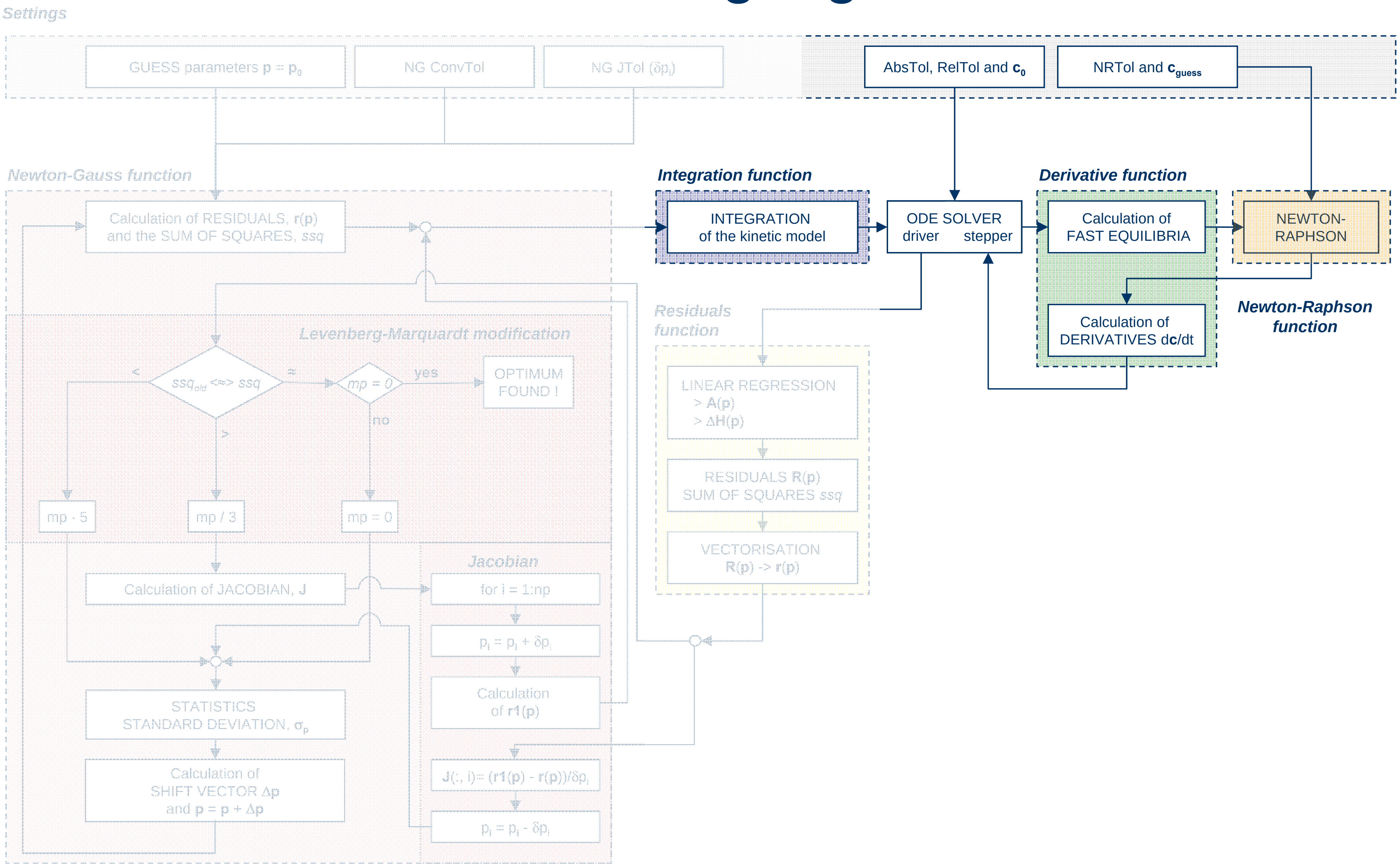
Residuals function

LINEAR REGRESSION
> $\mathbf{A}(\mathbf{p})$
> $\Delta\mathbf{H}(\mathbf{p})$

RESIDUALS $\mathbf{R}(\mathbf{p})$ SUM OF SQUARES *ssq*

VECTORISATION $\mathbf{R}(\mathbf{p})$ -> $\mathbf{r}(\mathbf{p})$

Kinetic modeling algorithm


Settings
GUESS parameters p = p_0
NG ConvTol
NG JTol (δp_i)
AbsTol, RelTol and $\mathbf{c_0}$
NRTol and $\mathbf{c_{guess}}$
Newton-Gauss function
Calculation of RESIDUALS, r(p) and the SUM OF SQUARES, *ssq*
Integration function
INTEGRATION of the kinetic model
ODE SOLVER driver stepper
Derivative function
Calculation of FAST EQUILIBRIA
NEWTON-RAPHSON
Calculation of DERIVATIVES d**c**/dt
Newton-Raphson function
Residuals function
Levenberg-Marquardt modification
ssq_{old} <≈> ssq
<
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
LINEAR REGRESSION
> A(p)
> ΔH(p)
RESIDUALS R(p)
SUM OF SQUARES *ssq*
VECTORISATION
R(p) -> r(p)
mp · 5
mp / 3
mp = 0
Jacobian
Calculation of JACOBIAN, J
for i = 1:np
$p_i = p_i + \delta p_i$
Calculation of r1(p)
STATISTICS
STANDARD DEVIATION, σ_p
J(:, i)= (r1(p) - r(p))/δp_i
Calculation of SHIFT VECTOR Δp and p = p + Δp
$p_i = p_i - \delta p_i$

Setup the kinetic model

$$A + B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

Setup the kinetic model

$$A + B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

$$\mathbf{X_r} = \begin{array}{c|cccc|c} & \multicolumn{4}{c}{\overleftrightarrow{\quad nc \quad}} & \\ & A & B & C & D & \\ \hline k_1 & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \updownarrow np \\ k_2 & \mathbf{0} & \mathbf{0} & \mathbf{2} & \mathbf{0} & \end{array}$$

Setup the kinetic model

$$A+B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

$$\mathbf{X}_\mathbf{r} = \begin{array}{c|cccc|c} & \multicolumn{4}{c|}{\longleftarrow nc \longrightarrow} & \\ & A & B & C & D & \\ \hline k_1 & 1 & 1 & 0 & 0 & \updownarrow \\ k_2 & 0 & 0 & 2 & 0 & np \end{array}$$

$$\mathbf{X}_\mathbf{p} = \begin{array}{c|cccc} & A & B & C & D \\ \hline k_1 & 0 & 0 & 1 & 0 \\ k_2 & 0 & 0 & 0 & 1 \end{array}$$

Setup the kinetic model

$$A + B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

$$\mathbf{X_r} = \begin{array}{c|cccc|c} & \multicolumn{4}{c}{\longleftarrow nc \longrightarrow} & \\ & A & B & C & D & \\ \hline k_1 & 1 & 1 & 0 & 0 & \updownarrow np \\ k_2 & 0 & 0 & 2 & 0 & \\ \hline \end{array}$$

$$\mathbf{X_p} = \begin{array}{c|cccc|} & A & B & C & D \\ \hline k_1 & 0 & 0 & 1 & 0 \\ k_2 & 0 & 0 & 0 & 1 \\ \hline \end{array}$$

$$\mathbf{X} = \mathbf{X_P} - \mathbf{X_R} = \begin{array}{c|cccc|} & A & B & C & D \\ \hline k_1 & -1 & -1 & 1 & 0 \\ k_2 & 0 & 0 & -2 & 1 \\ \hline \end{array}$$

Setup the kinetic model

$$A + B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

$$\mathbf{X_r} = \overset{nc}{\begin{array}{c|cccc} & A & B & C & D \\ \hline k_1 & 1 & 1 & 0 & 0 \\ k_2 & 0 & 0 & 2 & 0 \end{array}} \; np$$

$$\mathbf{X_p} = \begin{array}{c|cccc} & A & B & C & D \\ \hline k_1 & 0 & 0 & 1 & 0 \\ k_2 & 0 & 0 & 0 & 1 \end{array}$$

$$\mathbf{X} = \mathbf{X_P} - \mathbf{X_R} = \begin{array}{c|cccc} & A & B & C & D \\ \hline k_1 & -1 & -1 & 1 & 0 \\ k_2 & 0 & 0 & -2 & 1 \end{array}$$

Rate laws

$$r_j = k_j \prod_{i=1}^{nc} c_i^{\mathbf{X_r}\, j,i} \qquad for\ j = 1 : np$$

$$\boxed{\begin{array}{l} r_1 = k_1 \cdot [A]_t \cdot [B]_t \\ r_2 = k_2 \cdot [C]_t^2 \end{array}}$$

Setup the kinetic model

$$A + B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

$$\mathbf{X_r} = \begin{array}{c|cccc} & A & B & C & D \\ \hline k_1 & 1 & 1 & 0 & 0 \\ k_2 & 0 & 0 & 2 & 0 \end{array} \qquad \overset{nc}{\longleftrightarrow} \quad \updownarrow np$$

$$\mathbf{X_p} = \begin{array}{c|cccc} & A & B & C & D \\ \hline k_1 & 0 & 0 & 1 & 0 \\ k_2 & 0 & 0 & 0 & 1 \end{array}$$

$$\mathbf{X} = \mathbf{X_P} - \mathbf{X_R} = \begin{array}{c|cccc} & A & B & C & D \\ \hline k_1 & -1 & -1 & 1 & 0 \\ k_2 & 0 & 0 & -2 & 1 \end{array}$$

Rate laws

$$r_j = k_j \prod_{i=1}^{nc} c_i^{\mathbf{X}_{\mathbf{r}\,j,i}} \qquad for\ j = 1 : np$$

$$\boxed{\begin{aligned} r_1 &= k_1 \cdot [A]_t \cdot [B]_t \\ r_2 &= k_2 \cdot [C]_t^2 \end{aligned}}$$

Derivatives

$$\frac{dc_i}{dt} = \sum_{j=1}^{np} \mathbf{X}_{j,i} \cdot r_j \qquad for\ \ i = 1 : nc$$

$$\boxed{\begin{aligned} \frac{d[A]_t}{dt} &= \frac{d[B]_t}{dt} = -r_1 \\ \frac{d[C]_t}{dt} &= r_1 - 2 \cdot r_2 \\ \frac{d[D]_t}{dt} &= r_2 \end{aligned}}$$

Setup the kinetic model

$$A + B \xrightarrow{k_1} C$$
$$2C \xrightarrow{k_2} D$$

$$\mathbf{X_r} = \begin{array}{c|c|c|c|c|} \multicolumn{5}{c}{\overset{nc}{\longleftrightarrow}} \\ & A & B & C & D \\ \hline k_1 & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \hline k_2 & \mathbf{0} & \mathbf{0} & \mathbf{2} & \mathbf{0} \\ \hline \end{array} \updownarrow np$$

$$\mathbf{X_p} = \begin{array}{c|c|c|c|c|} & A & B & C & D \\ \hline k_1 & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \hline k_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \hline \end{array}$$

$$\mathbf{X} = \mathbf{X_P} - \mathbf{X_R} = \begin{array}{c|c|c|c|c|} & A & B & C & D \\ \hline k_1 & \mathbf{-1} & \mathbf{-1} & \mathbf{1} & \mathbf{0} \\ \hline k_2 & \mathbf{0} & \mathbf{0} & \mathbf{-2} & \mathbf{1} \\ \hline \end{array}$$

__Rate laws__

$$r_j = k_j \prod_{i=1}^{nc} c_i^{\mathbf{X_r}\,j,i} \qquad for\ j = 1 : np$$

$$\boxed{\begin{aligned} r_1 &= k_1 \cdot [A]_t \cdot [B]_t \\ r_2 &= k_2 \cdot [C]_t^2 \end{aligned}}$$

__Derivatives__

$$\frac{dc_i}{dt} = \sum_{j=1}^{np} \mathbf{X}_{j,i} \cdot r_j \qquad for\ \ i = 1 : nc$$

$$\boxed{\begin{aligned} \frac{d[A]_t}{dt} &= \frac{d[B]_t}{dt} = -r_1 \\ \frac{d[C]_t}{dt} &= r_1 - 2 \cdot r_2 \\ \frac{d[D]_t}{dt} &= r_2 \end{aligned}}$$

Most of the time : no analytical solution for this system of ODEs
→ **Numerical integration**

Dosing events

$$A+B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

In case of dosing, the set of ODEs is modified accordingly :

Dosing events

$$A + B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

In case of dosing, the set of ODEs is modified accordingly :

Rate laws

$$r_j = k_j \prod_{i=1}^{ns} c_i^{\mathbf{X_r}_{j,i}} \qquad for\ j = 1 : np$$

$$\boxed{\begin{array}{l} r_1 = k_1 \cdot [A]_t \cdot [B]_t \\ r_2 = k_2 \cdot [C]_t^2 \end{array}}$$

Dosing events

$$A + B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

In case of dosing, the set of ODEs is modified accordingly :

Rate laws

$$r_j = k_j \prod_{i=1}^{ns} c_i^{\mathbf{X}_{\mathbf{r}\; j,i}} \qquad for\ j = 1 : np$$

$$\boxed{\begin{aligned} r_1 &= k_1 \cdot [A]_t \cdot [B]_t \\ r_2 &= k_2 \cdot [C]_t^{\,2} \end{aligned}}$$

Derivatives

$$\frac{dc_i}{dt} = \left(\sum_{j=1}^{np} \mathbf{X}_{j,i} \cdot r_j \right) + \frac{F}{V_t} \cdot \left(c_i^{feed} - c_i \right)$$

$$\frac{dV}{dt} = F \qquad for\ \ i = 1 : nc$$

$$\boxed{\begin{aligned} \frac{d[A]_t}{dt} &= -r_1 + \frac{F}{V_t} \cdot \left([A]^{feed} - [A]_t \right) \\ \frac{d[B]_t}{dt} &= -r_1 + \frac{F}{V_t} \cdot \left([B]^{feed} - [B]_t \right) \\ \frac{d[C]_t}{dt} &= r_1 - 2 \cdot r_2 + \frac{F}{V_t} \cdot \left([C]^{feed} - [C]_t \right) \\ \frac{d[D]_t}{dt} &= r_2 + \frac{F}{V_t} \cdot \left([D]^{feed} - [D]_t \right) \\ \frac{dV}{dt} &= F \end{aligned}}$$

Dosing events

$$A + B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

In case of dosing, the set of ODEs is modified accordingly :

<u>*Rate laws*</u>

$$r_j = k_j \prod_{i=1}^{ns} c_i^{\mathbf{X_r}_{j,i}} \qquad for\ j = 1 : np$$

$$\boxed{\begin{aligned} r_1 &= k_1 \cdot [A]_t \cdot [B]_t \\ r_2 &= k_2 \cdot [C]_t^2 \end{aligned}}$$

<u>*Derivatives*</u>

$$\frac{dc_i}{dt} = \left(\sum_{j=1}^{np} \mathbf{X}_{j,i} \cdot r_j \right) + \frac{F}{V_t} \cdot \left(c_i^{feed} - c_i \right)$$

$$\frac{dV}{dt} = F \qquad for\ \ i = 1 : nc$$

$$\boxed{\begin{aligned} \frac{d[A]_t}{dt} &= -r_1 + \frac{F}{V_t} \cdot \left([A]^{feed} - [A]_t \right) \\ \frac{d[B]_t}{dt} &= -r_1 + \frac{F}{V_t} \cdot \left([B]^{feed} - [B]_t \right) \\ \frac{d[C]_t}{dt} &= r_1 - 2 \cdot r_2 + \frac{F}{V_t} \cdot \left([C]^{feed} - [C]_t \right) \\ \frac{d[D]_t}{dt} &= r_2 + \frac{F}{V_t} \cdot \left([D]^{feed} - [D]_t \right) \\ \frac{dV}{dt} &= F \end{aligned}}$$

The dosing event adds an additional ODE and a term to the ODEs of all species

NB : $+\frac{F}{V_t} \cdot A_{in}$ *= added material*

$-\frac{F}{V_t} \cdot A$ *= dilution phenomenon*

Numerical integration of the model

- First Approach : Euler’s method

$$c_i\left(t+\Delta t\right) \approx c_i\left(t\right) + \left(\frac{dc_i}{dt}\right)_t \cdot \Delta t$$

Numerical integration of the model

- First Approach : Euler’s method

$$c_i\left(t+\Delta t\right) \approx c_i\left(t\right) + \left(\frac{dc_i}{dt}\right)_t \cdot \Delta t$$

Applied to our specific example *without dosing*

$$A + B \xrightarrow{k_1} C$$

$$2C \xrightarrow{k_2} D$$

$$[C]_{t+\Delta t} \approx [C]_t + \left(\frac{d\,[C]_t}{dt}\right)_t \cdot \Delta t$$

$$= [C]_t + \left(k_1 \cdot [A]_t \cdot [B]_t - k_2 \cdot [C]_t{}^2\right) \cdot \Delta t$$

Numerical integration of the model

- First Approach : Euler’s method

$$c_i\left(t+\Delta t\right)\approx c_i\left(t\right)+\left(\frac{dc_i}{dt}\right)_t\cdot\Delta t$$

Applied to our specific example *without dosing*

$$A+B\xrightarrow{k_1}C$$
$$2C\xrightarrow{k_2}D$$

$$[C]_{t+\Delta t}\approx[C]_t+\left(\frac{d[C]_t}{dt}\right)_t\cdot\Delta t$$
$$=[C]_t+\left(k_1\cdot[A]_t\cdot[B]_t-k_2\cdot[C]_t^{\,2}\right)\cdot\Delta t$$

- Nowadays, more sophisticated integration methods exist (e.g. ode45) with a ***stepsize control***

Stepsize control and stiff problems

- In stepsize controlled ODE solvers, the stepsize is adjusted at each step to meet the user-specified accuracy

Stepsize control and stiff problems

- In stepsize controlled ODE solvers, the stepsize is adjusted at each step to meet the user-specified accuracy

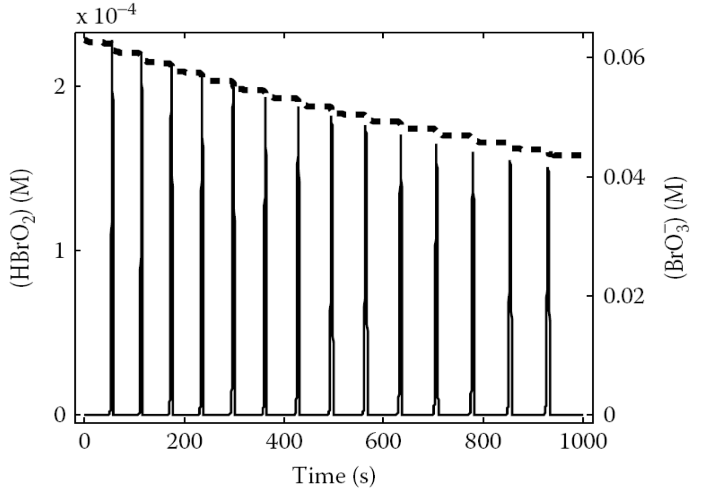
- The accuracy is measured with absolute (*AbsTol*) and relative (*RelTol*) tolerance’s values.

Stepsize control and stiff problems

- In stepsize controlled ODE solvers, the stepsize is adjusted at each step to meet the user-specified accuracy

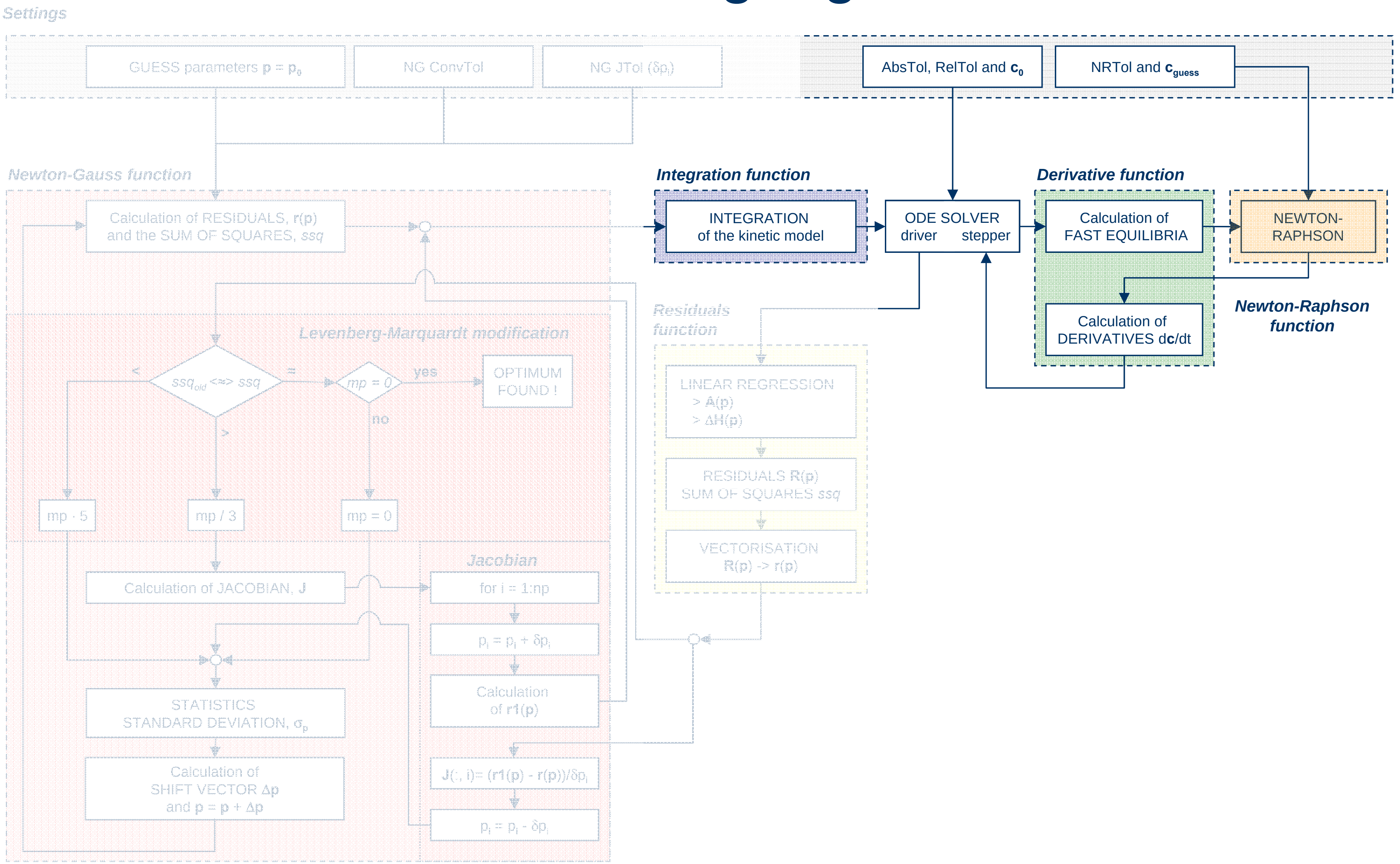
- The accuracy is measured with absolute (*AbsTol*) and relative (*RelTol*) tolerance's values.

- For some kinetic models, the slopes of the concentration profiles are dramatically different (stiff problem) and require the use of a **stiff ODE solver** (eg. ode15s)

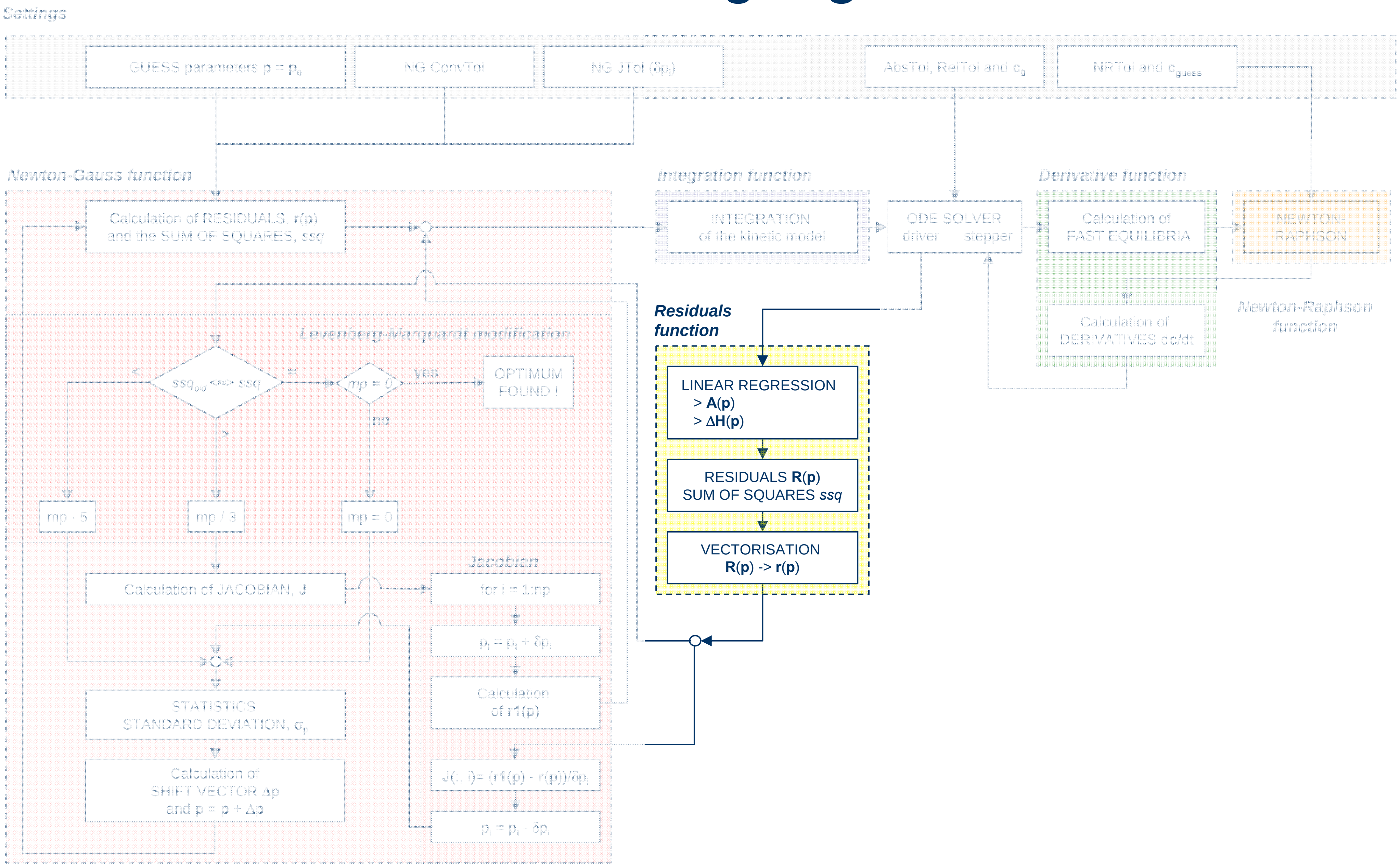


Gemperline, P. (2006). Practical Guide to Chemometrics (2nd edition), Taylor and Francis, Boca Raton, USA.

Kinetic modeling algorithm



Kinetic modeling algorithm



Settings

GUESS parameters $\mathbf{p} = \mathbf{p}_0$

NG ConvTol

NG JTol (δp_i)

AbsTol, RelTol and $\mathbf{c}_0$

NRTol and $\mathbf{c}_{guess}$

Newton-Gauss function

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

Levenberg-Marquardt modification

ssq_{old} <=> *ssq*

<

=

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

Jacobian

for i = 1:np

$p_i = p_i + \delta p_i$

Calculation of **r1**(**p**)

$J(:, i) = (\mathbf{r1}(\mathbf{p}) - \mathbf{r}(\mathbf{p}))/\delta p_i$

$p_i = p_i - \delta p_i$

Integration function

INTEGRATION of the kinetic model

ODE SOLVER driver stepper

Derivative function

Calculation of FAST EQUILIBRIA

Calculation of DERIVATIVES dc/dt

NEWTON-RAPHSON

Newton-Raphson function

Residuals function

LINEAR REGRESSION
> **A**(**p**)
> Δ**H**(**p**)

RESIDUALS **R**(**p**)
SUM OF SQUARES *ssq*

VECTORISATION
R(**p**) -> **r**(**p**)

Linear and Nonlinear parameters

GENERAL CONSIDERATION

Linear and Nonlinear parameters

GENERAL CONSIDERATION

- If *S*(**p**) is a measured signal depending on the parameters vector **p**

Linear parameters are defined as:

$$\left(\frac{\partial S(\mathbf{p})}{\partial p_i}\right)_{p_{j\neq i}} \neq f(p_i)$$

and **Nonlinear parameters** defined as:

$$\left(\frac{\partial S(\mathbf{p})}{\partial p_i}\right)_{p_{j\neq i}} = f(p_i)$$

Linear and Nonlinear parameters

GENERAL CONSIDERATION

- If $S(\mathbf{p})$ is a measured signal depending on the parameters vector **p**

Linear parameters are defined as:

$$\left(\frac{\partial S(\mathbf{p})}{\partial p_i}\right)_{p_{j\neq i}} \neq f(p_i)$$

and **Nonlinear parameters** defined as:

$$\left(\frac{\partial S(\mathbf{p})}{\partial p_i}\right)_{p_{j\neq i}} = f(p_i)$$

APPLIED TO KINETIC MODELING :

- **C** and **A** are LINEAR parameters with respect to **Y (Beer's law)**

$$\mathbf{Y} = \mathbf{C} \cdot \mathbf{A}$$

- **dξ_{mol}/dt** and **ΔH$_r$** are LINEAR parameters with respect to **q (Reaction heat balance)**

$$\mathbf{q} = \frac{\mathbf{d}\xi_{\mathbf{mol}}}{\mathbf{dt}} \cdot \left(-\Delta \mathbf{H}_{\mathbf{R}}\right)$$

- Rate constants are NONLINEAR parameters with respect to **C** and **dξ_{mol}/dt**
so are they for **Y** and for **q**

Separation of linear parameters – LINEAR REGRESSION

- At each iteration, the linear parameters are calculated in one step as the best linear estimate in the least squares sense

Separation of linear parameters – LINEAR REGRESSION

- At each iteration, the linear parameters are calculated in one step as the best linear estimate in the least squares sense

Pure spectra
(from Spectroscopy)

$$\mathbf{A} = \mathbf{C}^{+} \cdot \mathbf{Y} = \left(\mathbf{C}^{t} \cdot \mathbf{C}\right)^{-1} \cdot \mathbf{C}^{t} \cdot \mathbf{Y}$$

Enthalpies
(from Calorimetry)

$$\mathbf{\Delta H_R} = -\left(\frac{\mathbf{d\xi_{mol}}}{\mathbf{dt}}\right)^{+} \cdot \mathbf{q} = -\left(\left(\frac{\mathbf{d\xi_{mol}}}{\mathbf{dt}}\right)^{t} \cdot \left(\frac{\mathbf{d\xi_{mol}}}{\mathbf{dt}}\right)\right)^{-1} \cdot \left(\frac{\mathbf{d\xi_{mol}}}{\mathbf{dt}}\right)^{t} \cdot \mathbf{q}$$

Remarks :

- The above formula is a multidimensional linear regression in a matrix notation
- The *superscript* + is meant for the PSEUDO-INVERSE.
  As the matrices are not square, the inverse is not defined.

Separation of linear parameters – LINEAR REGRESSION

- At each iteration, the linear parameters are calculated in one step as the best linear estimate in the least squares sense

Pure spectra
(from Spectroscopy)

$$\mathbf{A} = \mathbf{C}^{+} \cdot \mathbf{Y} = \left(\mathbf{C}^{t} \cdot \mathbf{C}\right)^{-1} \cdot \mathbf{C}^{t} \cdot \mathbf{Y}$$

Enthalpies
(from Calorimetry)

$$\Delta\mathbf{H}_{\mathbf{R}} = -\left(\frac{\mathbf{d}\xi_{\mathbf{mol}}}{\mathbf{dt}}\right)^{+} \cdot \mathbf{q} = -\left(\left(\frac{\mathbf{d}\xi_{\mathbf{mol}}}{\mathbf{dt}}\right)^{t} \cdot \left(\frac{\mathbf{d}\xi_{\mathbf{mol}}}{\mathbf{dt}}\right)\right)^{-1} \cdot \left(\frac{\mathbf{d}\xi_{\mathbf{mol}}}{\mathbf{dt}}\right)^{t} \cdot \mathbf{q}$$

THIS LINEAR REGRESSION MAKES ANY CALIBRATION OF THE ABSORPTIVITIES REDUNDANT !

The Residuals and the sum of squares

- The residuals are defined as the difference between the measurement and the model **(matrix !)**

- By the sum of squares we mean the sum of all squared residuals **(scalar !)**

The Residuals and the sum of squares

- The residuals are defined as the difference between the measurement and the model **(matrix !)**

- By the sum of squares we mean the sum of all squared residuals **(scalar !)**

	Residuals	**Sum of squares**
Spectroscopy :	$\mathbf{R}_{\mathbf{spec}} = \mathbf{Y} - \mathbf{Y}_{\mathbf{calc}} = \mathbf{Y} - \mathbf{C} \cdot \mathbf{A}$	$ssq_{spec} = \sum_{i=1}^{nt} \sum_{j=1}^{nw} \mathbf{R}_{\mathbf{spec}}(i,j)$
Calorimetry :	$\mathbf{r}_{\mathbf{cal}} = \mathbf{q} - \mathbf{q}_{\mathbf{calc}} \quad = \mathbf{q} - \frac{d\xi_{mol}}{dt} \cdot (-\Delta H_R)$	$ssq_{cal} = \sum_{i=1}^{nt} \mathbf{r}_{\mathbf{cal}}(i)$

The Residuals and the sum of squares

- The residuals are defined as the difference between the measurement and the model **(matrix !)**

- By the sum of squares we mean the sum of all squared residuals **(scalar !)**

	Residuals	Sum of squares
Spectroscopy :	$\mathbf{R}_{\mathbf{spec}} = \mathbf{Y} - \mathbf{Y}_{\mathbf{calc}} = \mathbf{Y} - \mathbf{C} \cdot \mathbf{A}$	$ssq_{spec} = \sum_{i=1}^{nt} \sum_{j=1}^{nw} \mathbf{R}_{\mathbf{spec}}(i, j)$
Calorimetry :	$\mathbf{r}_{\mathbf{cal}} = \mathbf{q} - \mathbf{q}_{\mathbf{calc}} \quad = \mathbf{q} - \frac{d\xi_{mol}}{dt} \cdot (-\Delta H_R)$	$ssq_{cal} = \sum_{i=1}^{nt} \mathbf{r}_{\mathbf{cal}}(i)$

ADVANCED PROBLEM :

- Combination of the signals : $ssq_{total} = w \cdot ssq_{spec} + (1 - w) \cdot ssq_{cal} \quad , \quad w = ?$

Vectorisation (Unfolding)

Vectorisation (Unfolding)

- The residuals of Spectroscopy and Calorimetry do not have the same dimension
  (How to combine a matrix with a vector ?)

Vectorisation (Unfolding)

- The residuals of Spectroscopy and Calorimetry do not have the same dimension
  (How to combine a matrix with a vector ?)

AND

- For practical reasons in the
  Newton-Gauss algorithm (see later)

Vectorisation (Unfolding)

- The residuals of Spectroscopy and Calorimetry do not have the same dimension
  (How to combine a matrix with a vector ?)

AND

- For practical reasons in the
  Newton-Gauss algorithm (see later)

$\Downarrow$

VECTORISATION :

R_{spec} and r_{cal} are unfolded into a long column vector **r**

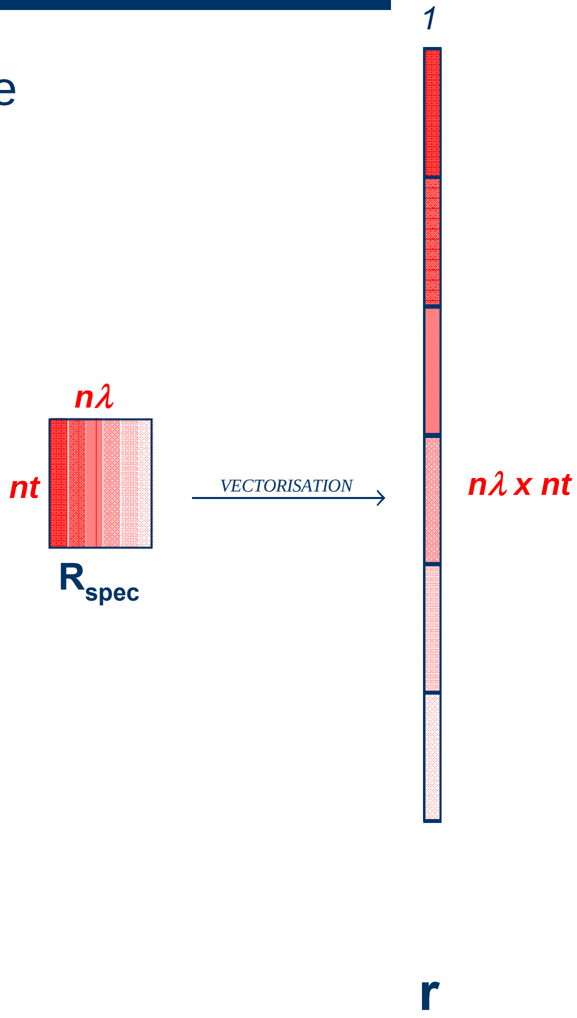
Vectorisation (Unfolding)

- The residuals of Spectroscopy and Calorimetry do not have the same dimension
  (How to combine a matrix with a vector ?)

AND

- For practical reasons in the Newton-Gauss algorithm (see later)

⇓



VECTORISATION :

$\mathbf{R}_{spec}$ **and** $\mathbf{r}_{cal}$ are unfolded into a long column vector **r**

Vectorisation (Unfolding)

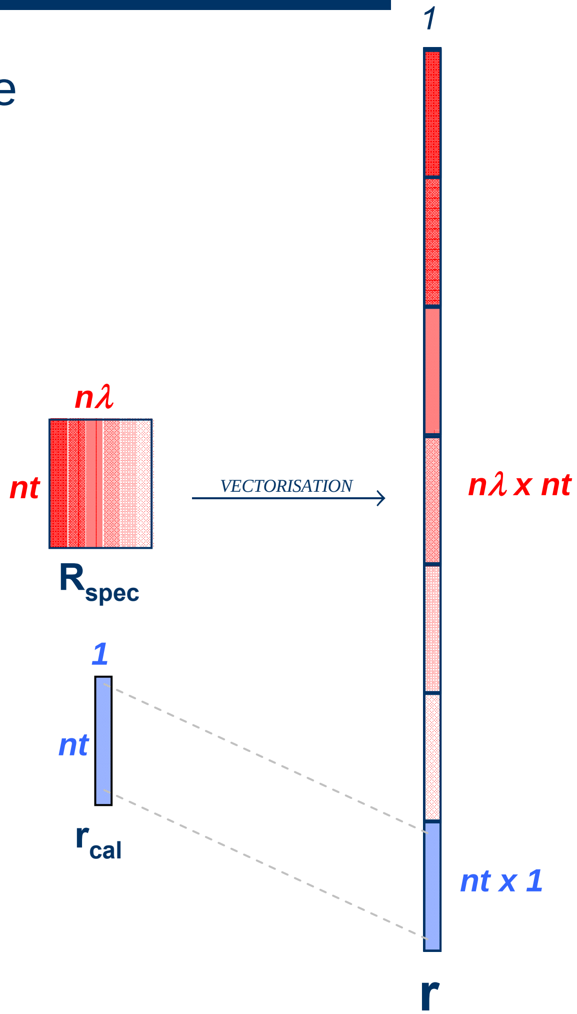
- The residuals of Spectroscopy and Calorimetry do not have the same dimension
  (How to combine a matrix with a vector ?)

AND

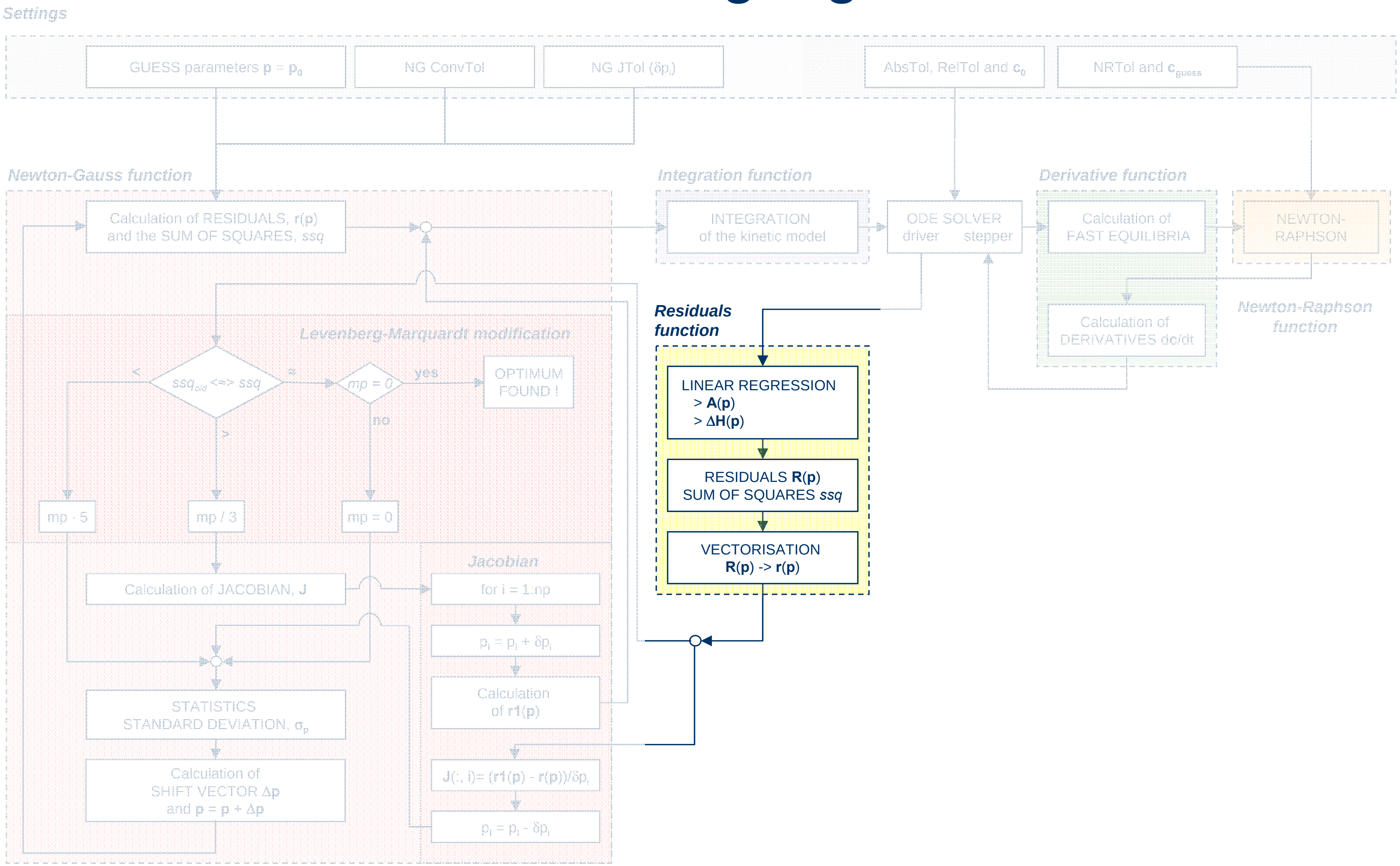
- For practical reasons in the Newton-Gauss algorithm (see later)

$\Downarrow$

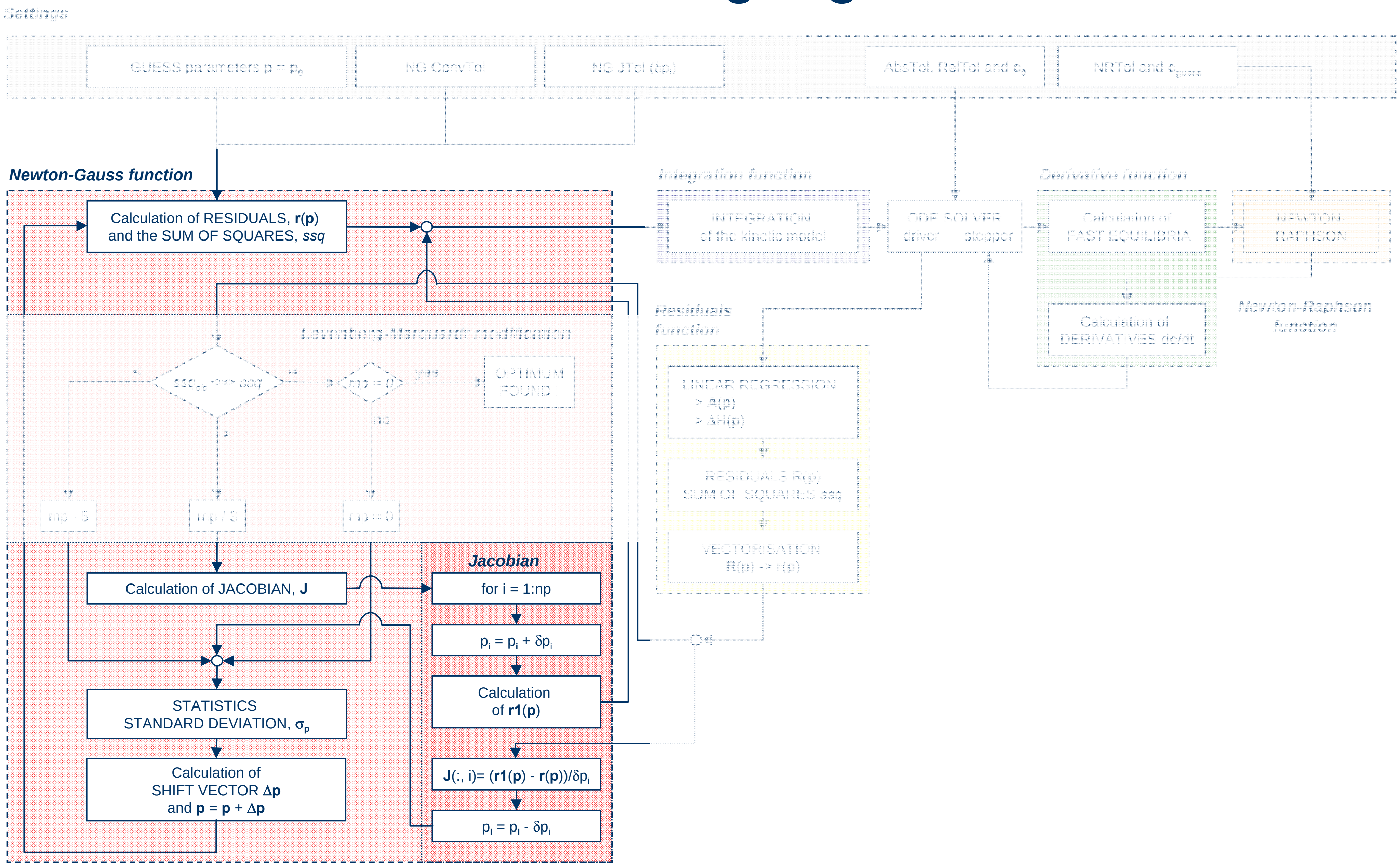
VECTORISATION :
R_{spec} and r_{cal} are unfolded into a long column vector **r**



Kinetic modeling algorithm


Settings
GUESS parameters $p = p_0$
NG ConvTol
NG JTol (δp_i)
AbsTol, RelTol and c_0
NRTol and c_{guess}
Newton-Gauss function
Calculation of RESIDUALS, r(p) and the SUM OF SQUARES, ssq
Integration function
INTEGRATION of the kinetic model
ODE SOLVER driver stepper
Derivative function
Calculation of FAST EQUILIBRIA
NEWTON-RAPHSON
Calculation of DERIVATIVES dc/dt
Newton-Raphson function
Residuals function
LINEAR REGRESSION > A(p) > ΔH(p)
RESIDUALS R(p) SUM OF SQUARES ssq
VECTORISATION R(p) -> r(p)
Levenberg-Marquardt modification
ssq_{old} <=> ssq
<
=
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, J
Jacobian
for i = 1:np
$p_i = p_i + \delta p_i$
Calculation of r1(p)
$J(:, i)= (r1(p) - r(p))/\delta p_i$
$p_i = p_i - \delta p_i$
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δp and p = p + Δp

Kinetic modeling algorithm



Settings

GUESS parameters $\mathbf{p} = \mathbf{p}_0$

NG ConvTol

NG JTol (δp_i)

AbsTol, RelTol and $\mathbf{c}_0$

NRTol and $\mathbf{c}_{guess}$

Newton-Gauss function

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

Levenberg-Marquardt modification

ssq_{clc} <=> ssq

<

≈

>

$mp = 0$

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, $\sigma_{\mathbf{p}}$

Calculation of SHIFT VECTOR $\Delta\mathbf{p}$ and $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$

Jacobian

for i = 1:np

$p_i = p_i + \delta p_i$

Calculation of **r1**(**p**)

J(:, i)= (**r1**(**p**) - **r**(**p**))/δp_i

$p_i = p_i - \delta p_i$

Integration function

INTEGRATION of the kinetic model

ODE SOLVER driver stepper

Derivative function

Calculation of FAST EQUILIBRIA

Calculation of DERIVATIVES dc/dt

NEWTON-RAPHSON

Newton-Raphson function

Residuals function

LINEAR REGRESSION
> **A**(**p**)
> ΔH(**p**)

RESIDUALS **R**(**p**) SUM OF SQUARES *ssq*

VECTORISATION **R**(**p**) -> **r**(**p**)

The Newton-Gauss algorithm

- To find the direction towards the minimum, the residuals are approximated by a **Taylor series expansion truncated after the first derivative**

The Newton-Gauss algorithm

- To find the direction towards the minimum, the residuals are approximated by a **Taylor series expansion truncated after the first derivative**

$$\mathbf{r}(\mathbf{p}+\Delta\mathbf{p}) = \mathbf{r}(\mathbf{p}) + \frac{\partial \mathbf{r}(\mathbf{p})}{\partial \mathbf{p}} \cdot \Delta\mathbf{p}$$

The Newton-Gauss algorithm

- To find the direction towards the minimum, the residuals are approximated by a **Taylor series expansion truncated after the first derivative**

$$\mathbf{r}(\mathbf{p}+\Delta\mathbf{p}) = \mathbf{r}(\mathbf{p}) + \frac{\partial\mathbf{r}(\mathbf{p})}{\partial\mathbf{p}}\cdot\Delta\mathbf{p}$$

$\Downarrow$ *Rearranging for* **r**(**p**)

$$\mathbf{r}(\mathbf{p}) \qquad = -\mathbf{J}\cdot\Delta\mathbf{p} + \mathbf{r}(\mathbf{p}+\Delta\mathbf{p})$$

The Newton-Gauss algorithm

- To find the direction towards the minimum, the residuals are approximated by a **Taylor series expansion truncated after the first derivative**

$$\mathbf{r}(\mathbf{p}+\Delta\mathbf{p}) = \mathbf{r}(\mathbf{p}) + \frac{\partial\mathbf{r}(\mathbf{p})}{\partial\mathbf{p}}\cdot\Delta\mathbf{p}$$

$$\Downarrow \quad \textit{Rearranging for } \mathbf{r}(\mathbf{p})$$

$$\mathbf{r}(\mathbf{p}) \qquad = -\mathbf{J}\cdot\Delta\mathbf{p} + \boxed{\mathbf{r}(\mathbf{p}+\Delta\mathbf{p})} \quad \approx 0\ !$$

The Newton-Gauss algorithm

- To find the direction towards the minimum, the residuals are approximated by a **Taylor series expansion truncated after the first derivative**

$$\mathbf{r}(\mathbf{p}+\Delta\mathbf{p}) = \mathbf{r}(\mathbf{p}) + \frac{\partial \mathbf{r}(\mathbf{p})}{\partial \mathbf{p}} \cdot \Delta\mathbf{p}$$

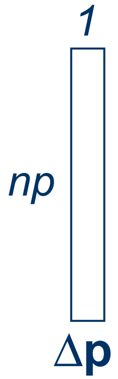
$\Downarrow$ *Rearranging for* **r**(**p**)

$$\mathbf{r}(\mathbf{p}) = -\mathbf{J}\cdot\Delta\mathbf{p} + \mathbf{r}(\mathbf{p}+\Delta\mathbf{p}) \quad \approx \mathbf{0}\ !$$

$\Downarrow$ *Linear regression to minimize* **r**(**p**+Δ**p**) *and rearranging for* Δ**p**

$$\boxed{\Delta\mathbf{p} = -\mathbf{J}^{+}\cdot\mathbf{r}(\mathbf{p})}$$

The **SHIFT VECTOR** is added to **p** for the next iteration



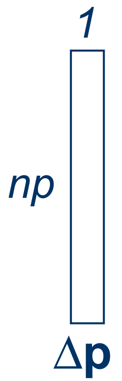
The Newton-Gauss algorithm

- To find the direction towards the minimum, the residuals are approximated by a **Taylor series expansion truncated after the first derivative**

$$\mathbf{r}(\mathbf{p}+\Delta\mathbf{p}) = \mathbf{r}(\mathbf{p}) + \frac{\partial\mathbf{r}(\mathbf{p})}{\partial\mathbf{p}} \cdot \Delta\mathbf{p}$$

$\Downarrow$ *Rearranging for* **r**(**p**)

$$\mathbf{r}(\mathbf{p}) = -\mathbf{J}\cdot\Delta\mathbf{p} + \boxed{\mathbf{r}(\mathbf{p}+\Delta\mathbf{p})} \approx \mathbf{0}\ \mathbf{!}$$



$\Downarrow$ *Linear regression to minimize* **r**(**p**+Δ**p**) *and rearranging for* **Δp**

$$\boxed{\Delta\mathbf{p} = \boxed{-\mathbf{J}^{+}}\,\mathbf{r}(\mathbf{p})}$$

The **SHIFT VECTOR** is added to **p** for the next iteration

The Newton-Gauss algorithm
requires the calculation of the Jacobian

$$\mathbf{J} = \frac{\partial\mathbf{r}(\mathbf{p})}{\partial\mathbf{p}}$$

The Jacobian

$$\mathbf{J} = \frac{\partial \mathbf{r}(\mathbf{p})}{\partial \mathbf{p}}$$

The Jacobian is a derivative of a matrix with respect to a vector

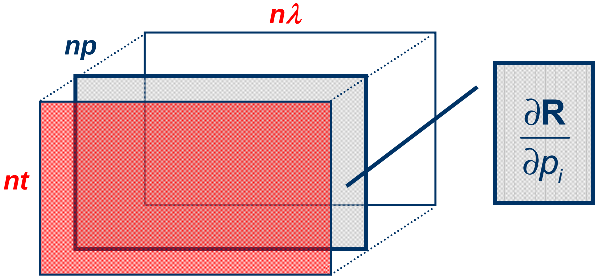
The Jacobian

$$\mathbf{J} = \frac{\partial \mathbf{r}(\mathbf{p})}{\partial \mathbf{p}}$$

The Jacobian is a derivative of a matrix with respect to a vector

Without vectorisation

(only spectroscopy)



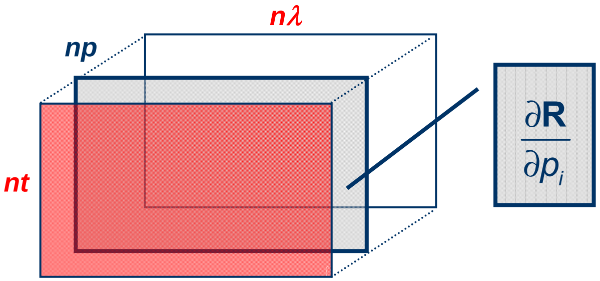
The Jacobian

$$\mathbf{J} = \frac{\partial \mathbf{r}(\mathbf{p})}{\partial \mathbf{p}}$$

The Jacobian is a derivative of a matrix with respect to a vector

Without vectorisation

(only spectroscopy)



TENSOR ☹

The Jacobian

$$\mathbf{J} = \frac{\partial \mathbf{r}(\mathbf{p})}{\partial \mathbf{p}}$$

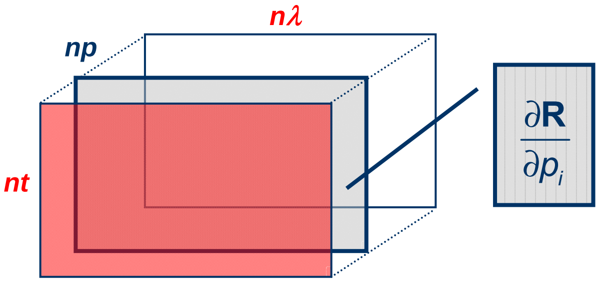
The Jacobian is a derivative of a matrix with respect to a vector

⇒

To compute the Jacobian, one needs to vectorise the residuals **R** into a long vector **r**

Without vectorisation

(only spectroscopy)



TENSOR ☹

The Jacobian

$$\mathbf{J} = \frac{\partial \mathbf{r}(\mathbf{p})}{\partial \mathbf{p}}$$

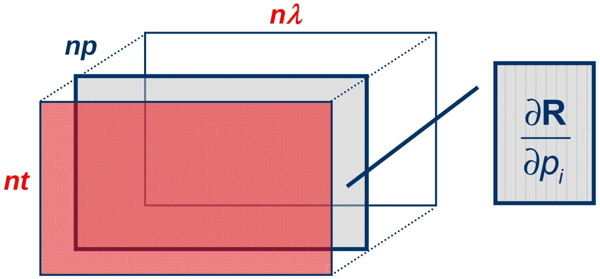
The Jacobian is a derivative of a matrix with respect to a vector

⇒

To compute the Jacobian, one needs to vectorise the residuals **R** into a long vector **r**

Without vectorisation

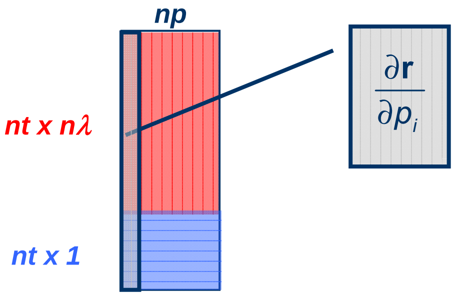
(only spectroscopy)



TENSOR ☹

With vectorisation

(Spectroscopy + Calorimetry)



The Jacobian

$$\mathbf{J} = \frac{\partial \mathbf{r}(\mathbf{p})}{\partial \mathbf{p}}$$

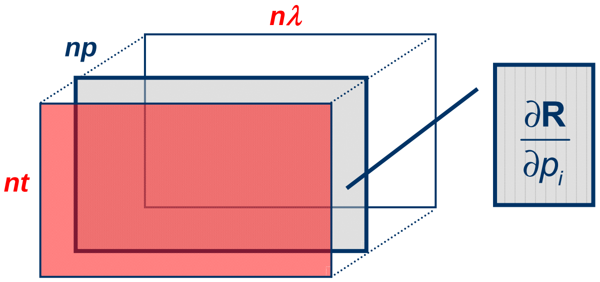
The Jacobian is a derivative of a matrix with respect to a vector

⇒

To compute the Jacobian, one needs to vectorise the residuals **R** into a long vector **r**

Without vectorisation

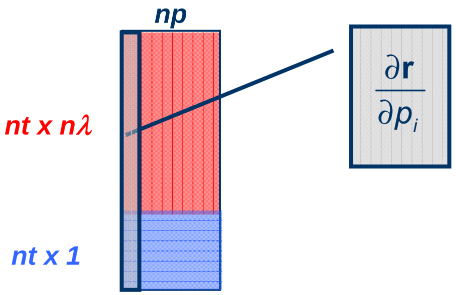
(only spectroscopy)



TENSOR ☹

With vectorisation

(Spectroscopy + Calorimetry)



MATRIX ☺

The Jacobian

$$\mathbf{J} = \frac{\partial \mathbf{r}(\mathbf{p})}{\partial \mathbf{p}}$$

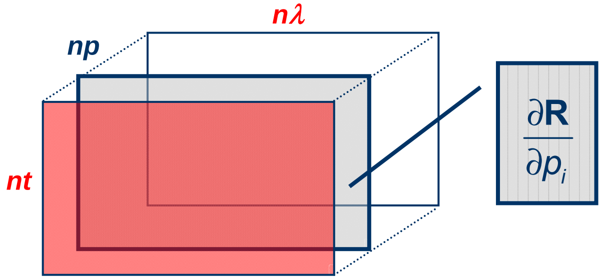
The Jacobian is a derivative of a matrix with respect to a vector

⇒

To compute the Jacobian, one needs to vectorise the residuals **R** into a long vector **r**

Without vectorisation

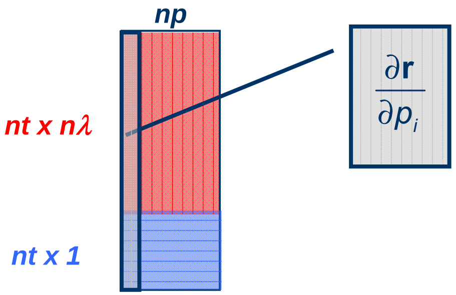
(only spectroscopy)



TENSOR ☹

With vectorisation

(Spectroscopy + Calorimetry)



MATRIX ☺

The Jacobian is computed using a forward finite difference

$$\frac{\partial \mathbf{r}(\mathbf{p})}{\partial p_i} = \frac{\mathbf{r}(\mathbf{p}+\delta p_i)-\mathbf{r}(\mathbf{p})}{\delta p_i} \quad \text{with } \delta p_i \approx 10^{-6} \cdot p_i$$

The Hessian (statistics)

$$\mathbf{H} = \mathbf{J}^{\mathbf{t}} \cdot \mathbf{J}$$

- Knowing that $\mathbf{H} = \mathbf{J}^{\mathbf{t}} \cdot \mathbf{J}$, the shift vector can be re-written as:

$$\Delta\mathbf{p} = -\mathbf{H}^{-1} \cdot \mathbf{J}^{\mathbf{t}} \cdot \mathbf{r}\left(\mathbf{p}\right)$$

The Hessian (statistics)

$\mathbf{H} = \mathbf{J}^{t} \cdot \mathbf{J}$

- Knowing that $\mathbf{H} = \mathbf{J}^{t} \cdot \mathbf{J}$, the shift vector can be re-written as:

$$\Delta\mathbf{p} = -\mathbf{H}^{-1} \cdot \mathbf{J}^{t} \cdot \mathbf{r}\left(\mathbf{p}\right)$$

The Hessian **H** is a square matrix (*np* x *np*) and
is the inverse of the variance/covariance matrix of **p** !

The Hessian (statistics)

$$\mathbf{H} = \mathbf{J}^{t}\cdot\mathbf{J}$$

- Knowing that $\mathbf{H} = \mathbf{J}^{t}\cdot\mathbf{J}$ the shift vector can be re-written as:

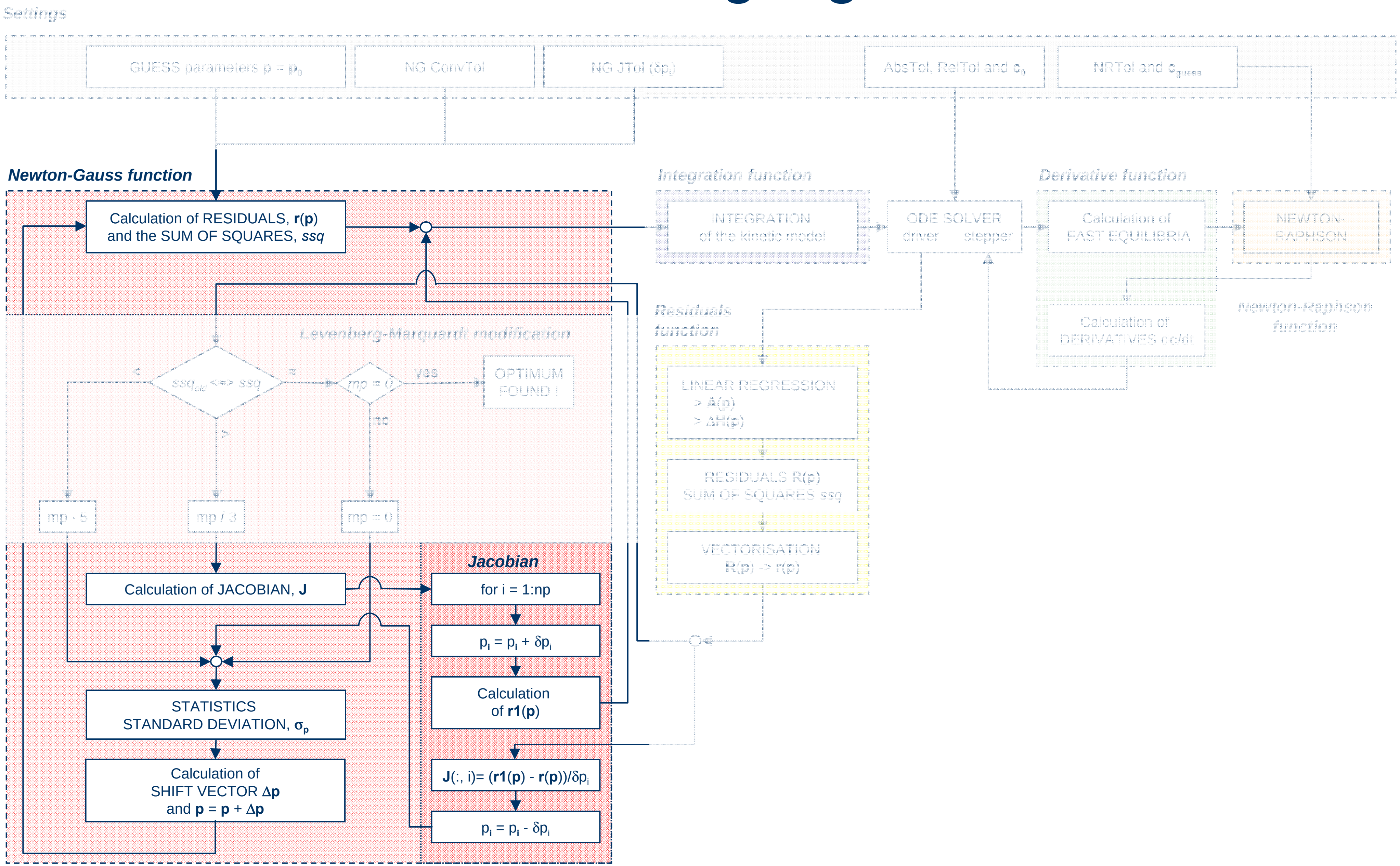
$$\Delta\mathbf{p} = -\mathbf{H}^{-1}\cdot\mathbf{J}^{t}\cdot\mathbf{r}(\mathbf{p})$$

The Hessian **H** is a square matrix (*np* x *np*) and
is the inverse of the variance/covariance matrix of **p** !

This allows the calculation of the
Standard Error of each parameter :

$$\sigma_{\mathbf{p}} = \sigma_{\mathbf{r}}\cdot\sqrt{diag\left(H^{-1}\right)} \qquad with \quad \sigma_{\mathbf{r}} = \sqrt{\frac{ssq}{df}} \approx \sigma_{\mathbf{Y}}$$

Kinetic modeling algorithm



Settings

GUESS parameters $\mathbf{p} = \mathbf{p}_0$

NG ConvTol

NG JTol (δp_i)

AbsTol, RelTol and $\mathbf{c}_0$

NRTol and $\mathbf{c}_{guess}$

Newton-Gauss function

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

Levenberg-Marquardt modification

ssq_{old} <≈> ssq

<

≈

>

$mp = 0$

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR $\Delta\mathbf{p}$ and $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$

Jacobian

for i = 1:np

$p_i = p_i + \delta p_i$

Calculation of **r1**(**p**)

J(:, i)= (**r1**(**p**) - **r**(**p**))/δp_i

$p_i = p_i - \delta p_i$

Integration function

INTEGRATION of the kinetic model

ODE SOLVER driver stepper

Derivative function

Calculation of FAST EQUILIBRIA

Calculation of DERIVATIVES dc/dt

NEWTON-RAPHSON

Newton-Raphson function

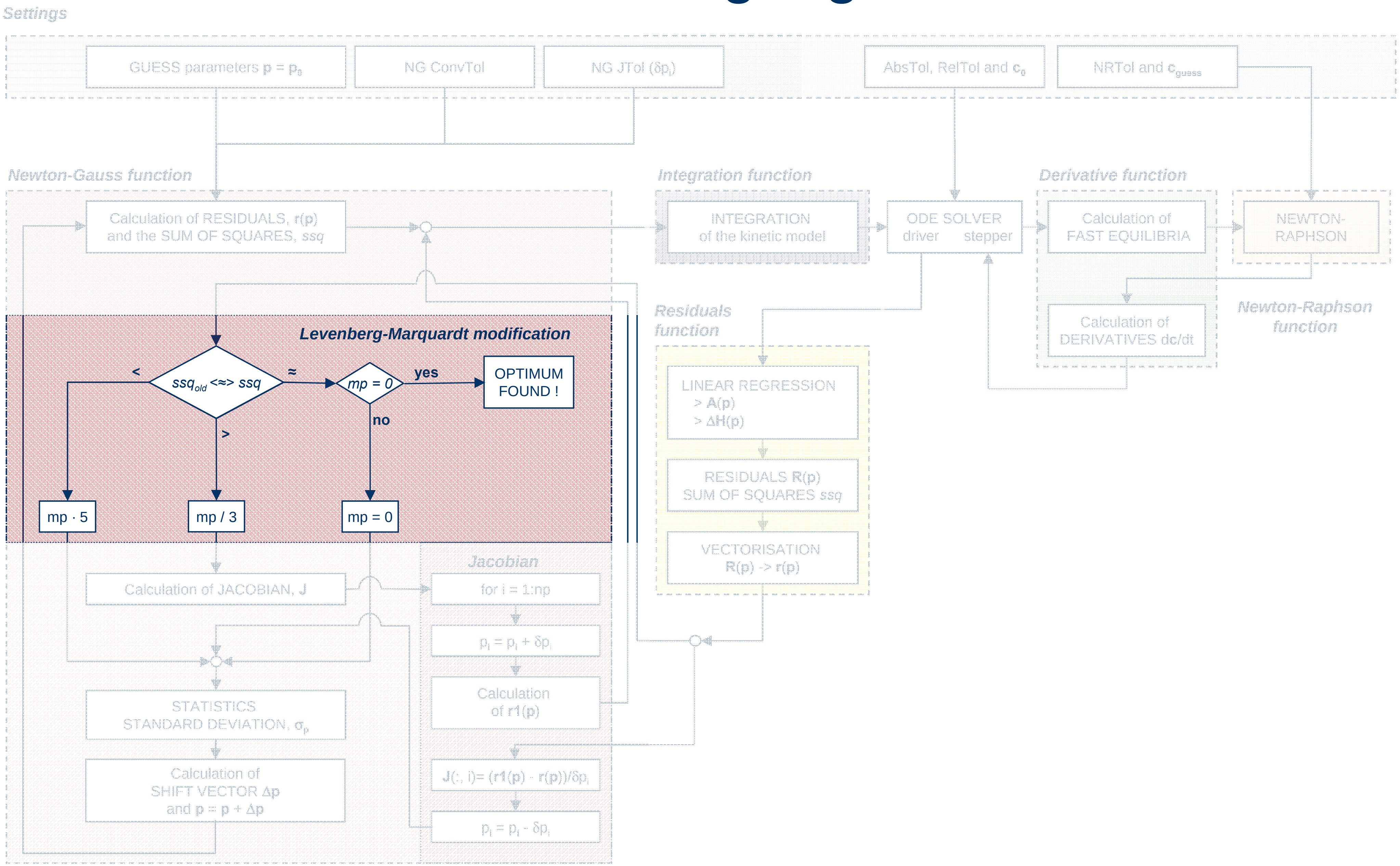
Residuals function

LINEAR REGRESSION
> **A**(**p**)
> Δ**H**(**p**)

RESIDUALS **R**(**p**) SUM OF SQUARES *ssq*

VECTORISATION **R**(**p**) -> **r**(**p**)

Kinetic modeling algorithm



Divergence in the NG algorithm

PROBLEM : The NG algorithm **DIVERGES** if the Taylor series expansion is not a good approximation for the residuals function (eg. poor initial guesses)

Divergence in the NG algorithm

PROBLEM : The NG algorithm **DIVERGES** if the Taylor series expansion is not a good approximation for the residuals function (eg. poor initial guesses)

SOLUTION : Do not use a Taylor series expansion but move in the steepest direction (opposite direction given by the Jacobian)

Divergence in the NG algorithm

PROBLEM : The NG algorithm **DIVERGES** if the Taylor series expansion is not a good approximation for the residuals function (eg. poor initial guesses)

SOLUTION : Do not use a Taylor series expansion but move in the steepest direction (opposite direction given by the Jacobian)

$$\Delta\mathbf{p} = -\mathbf{H}^{-1} \cdot \mathbf{J}^{t} \cdot \mathbf{r}(\mathbf{p})$$

Inverse Hessian method
(Newton-Gauss)

Is there a way to switch progressively from one method to the other ?

$$\Delta\mathbf{p} = -\mathbf{J}^{t} \cdot \mathbf{r}(\mathbf{p})$$

Steepest Descent Method

Divergence in the NG algorithm

PROBLEM : The NG algorithm **DIVERGES** if the Taylor series expansion is not a good approximation for the residuals function (eg. poor initial guesses)

SOLUTION : Do not use a Taylor series expansion but move in the steepest direction (opposite direction given by the Jacobian)

$$\Delta\mathbf{p} = -\mathbf{H}^{-1} \cdot \mathbf{J}^{\mathbf{t}} \cdot \mathbf{r}(\mathbf{p})$$

Inverse Hessian method
(Newton-Gauss)

$$\boxed{\Delta\mathbf{p} = -\left(\mathbf{H} + mp \cdot \mathbf{I}\right)^{-1} \cdot \mathbf{J}^{\mathbf{t}} \cdot \mathbf{r}(\mathbf{p})}$$

Levenberg-Marquardt
modification

$$\Delta\mathbf{p} = -\mathbf{J}^{\mathbf{t}} \cdot \mathbf{r}(\mathbf{p})$$

Steepest Descent Method

Divergence in the NG algorithm

PROBLEM : The NG algorithm **DIVERGES** if the Taylor series expansion is not a good approximation for the residuals function (eg. poor initial guesses)

SOLUTION : Do not use a Taylor series expansion but move in the steepest direction (opposite direction given by the Jacobian)

$$\Delta \mathbf{p} = -\mathbf{H}^{-1} \cdot \mathbf{J}^{t} \cdot \mathbf{r}(\mathbf{p})$$

Inverse Hessian method
(Newton-Gauss)

$$\boxed{\Delta \mathbf{p} = -\left(\mathbf{H} + mp \cdot \mathbf{I}\right)^{-1} \cdot \mathbf{J}^{t} \cdot \mathbf{r}(\mathbf{p})}$$

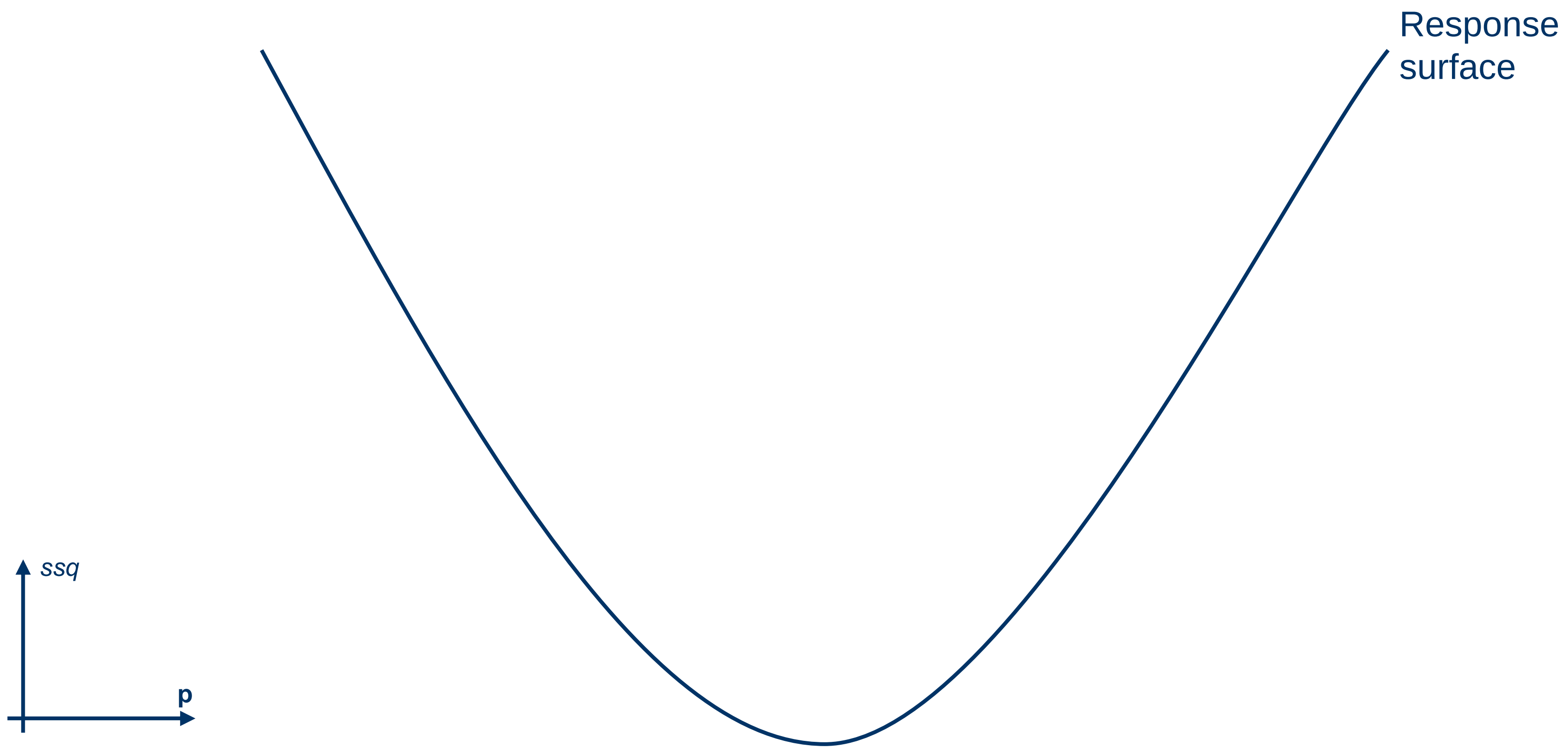
Levenberg-Marquardt
modification

The Marquardt parameter (*mp*) is a scalar added to the diagonal elements of **H** to decrease its influence on Δ**p** and shorten the magnitude of Δ**p**

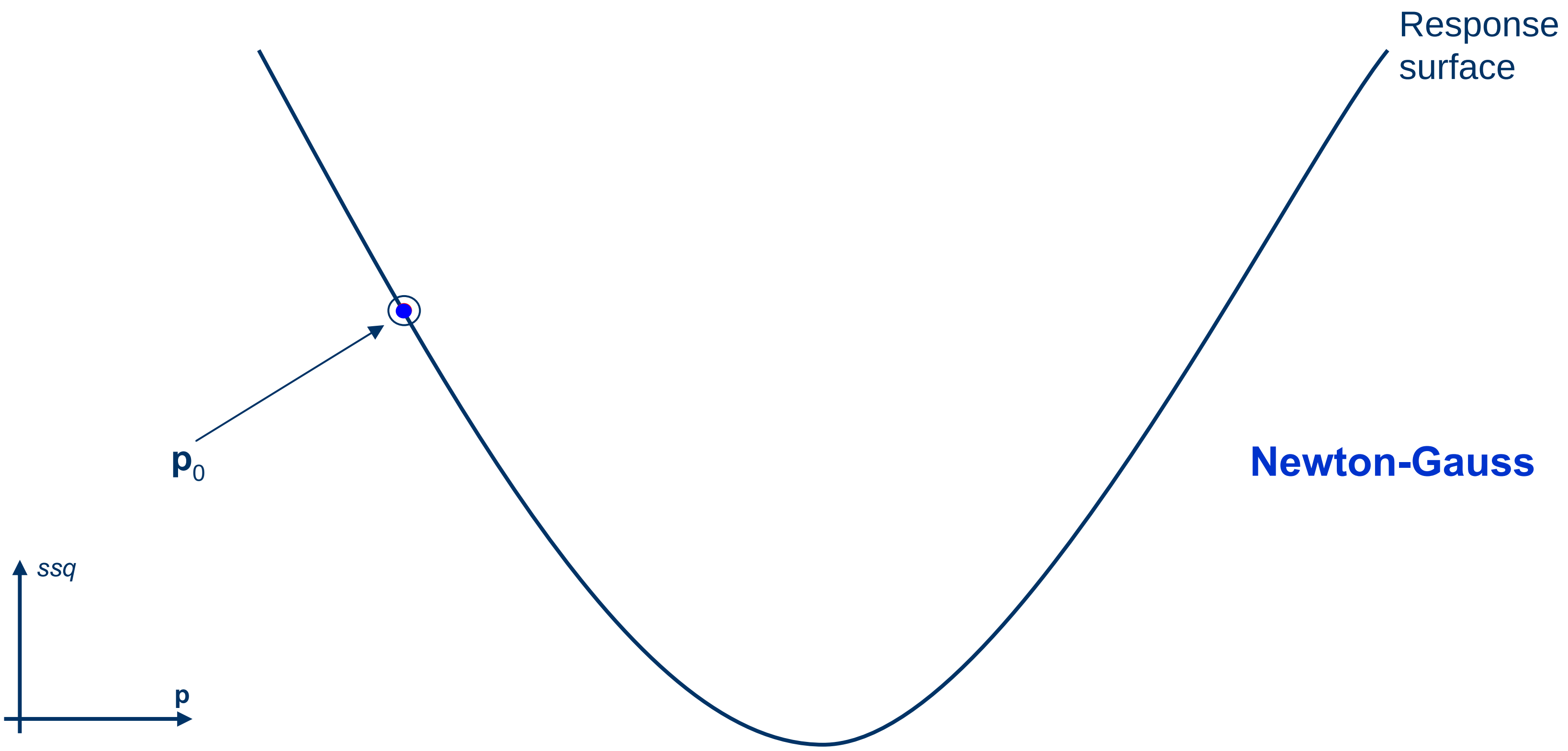
$$\Delta \mathbf{p} = -\mathbf{J}^{t} \cdot \mathbf{r}(\mathbf{p})$$

Steepest Descent Method

Geometrical interpretation on the response surface

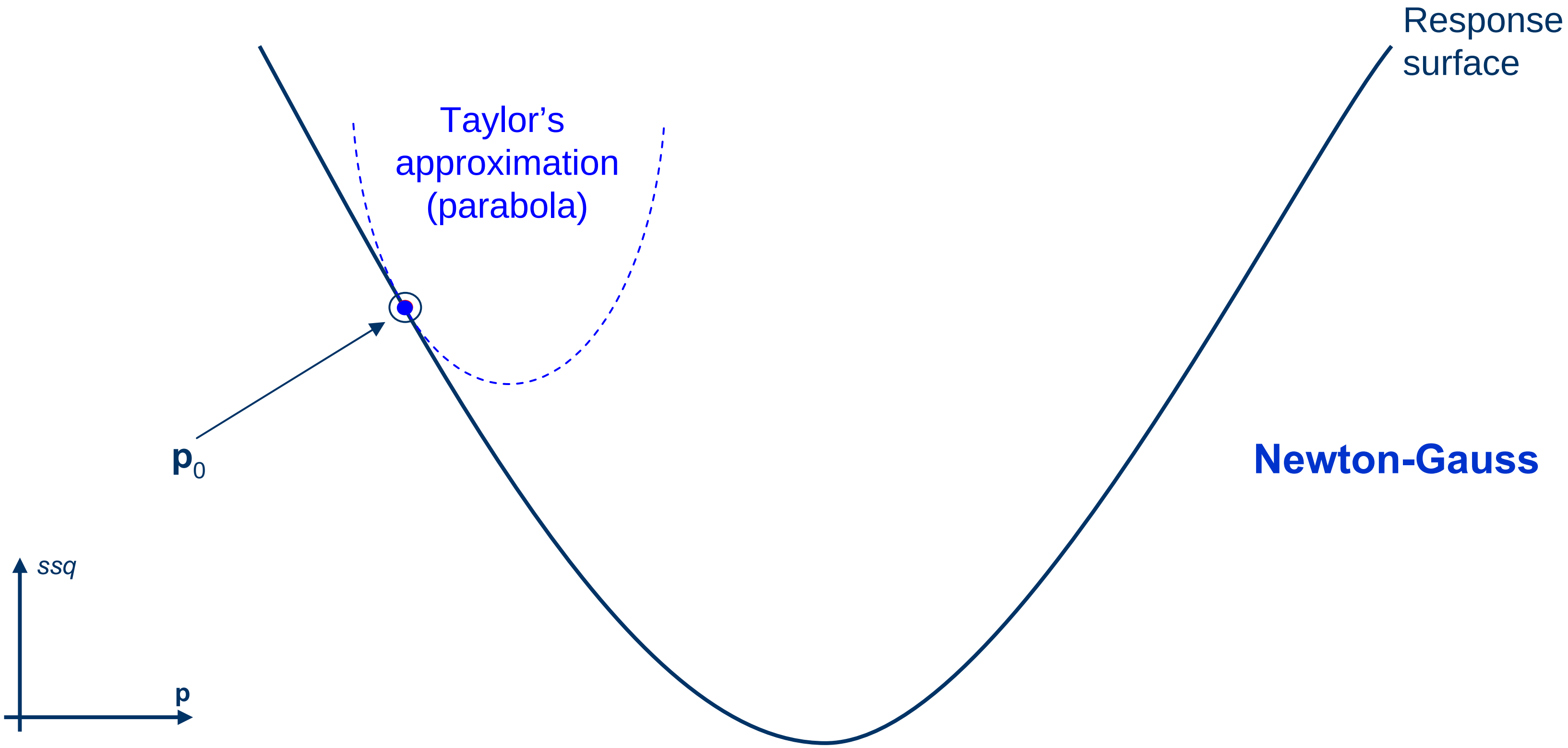


Geometrical interpretation on the response surface



Newton-Gauss

Geometrical interpretation on the response surface



Response surface

Taylor's approximation (parabola)

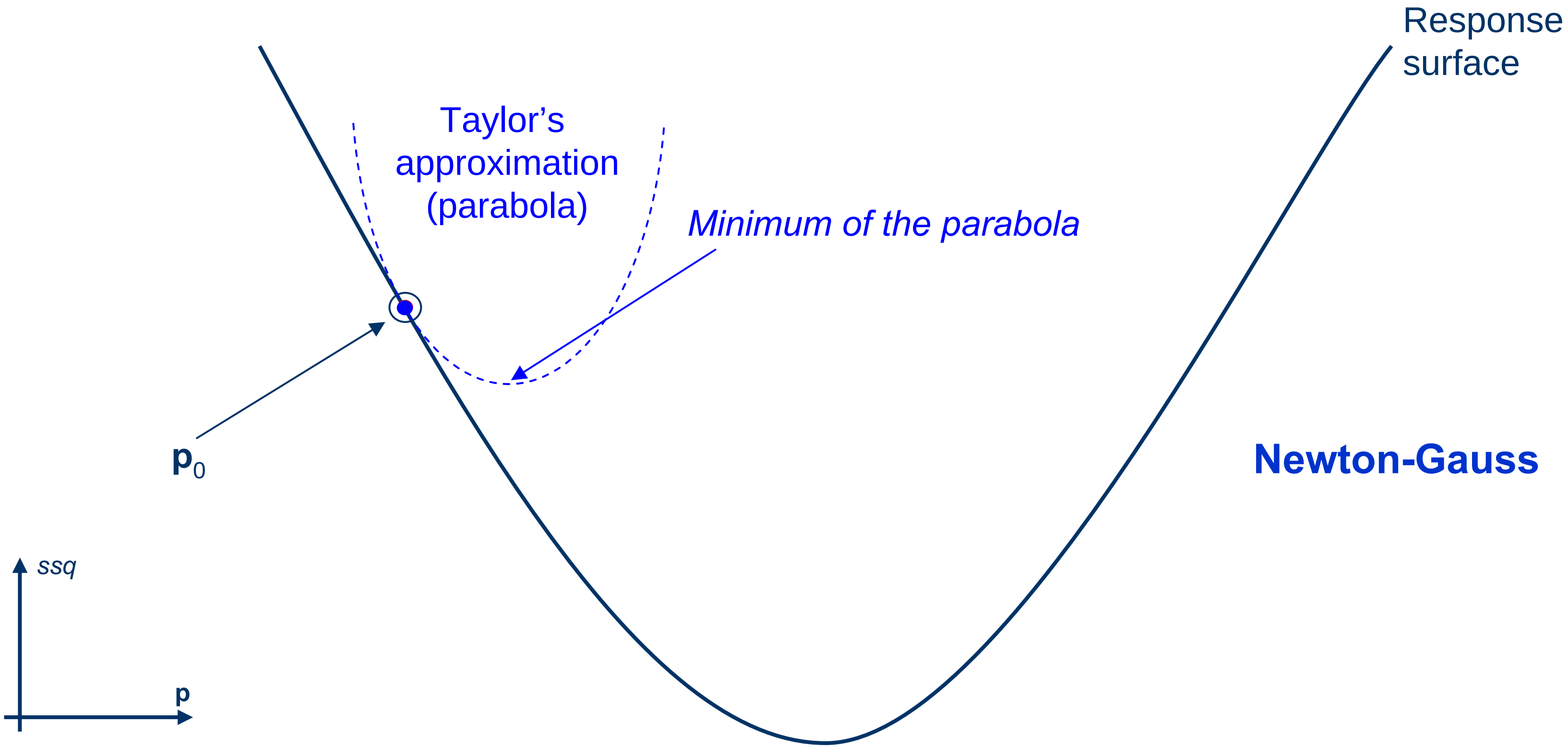
$\mathbf{p}_0$

ssq

p

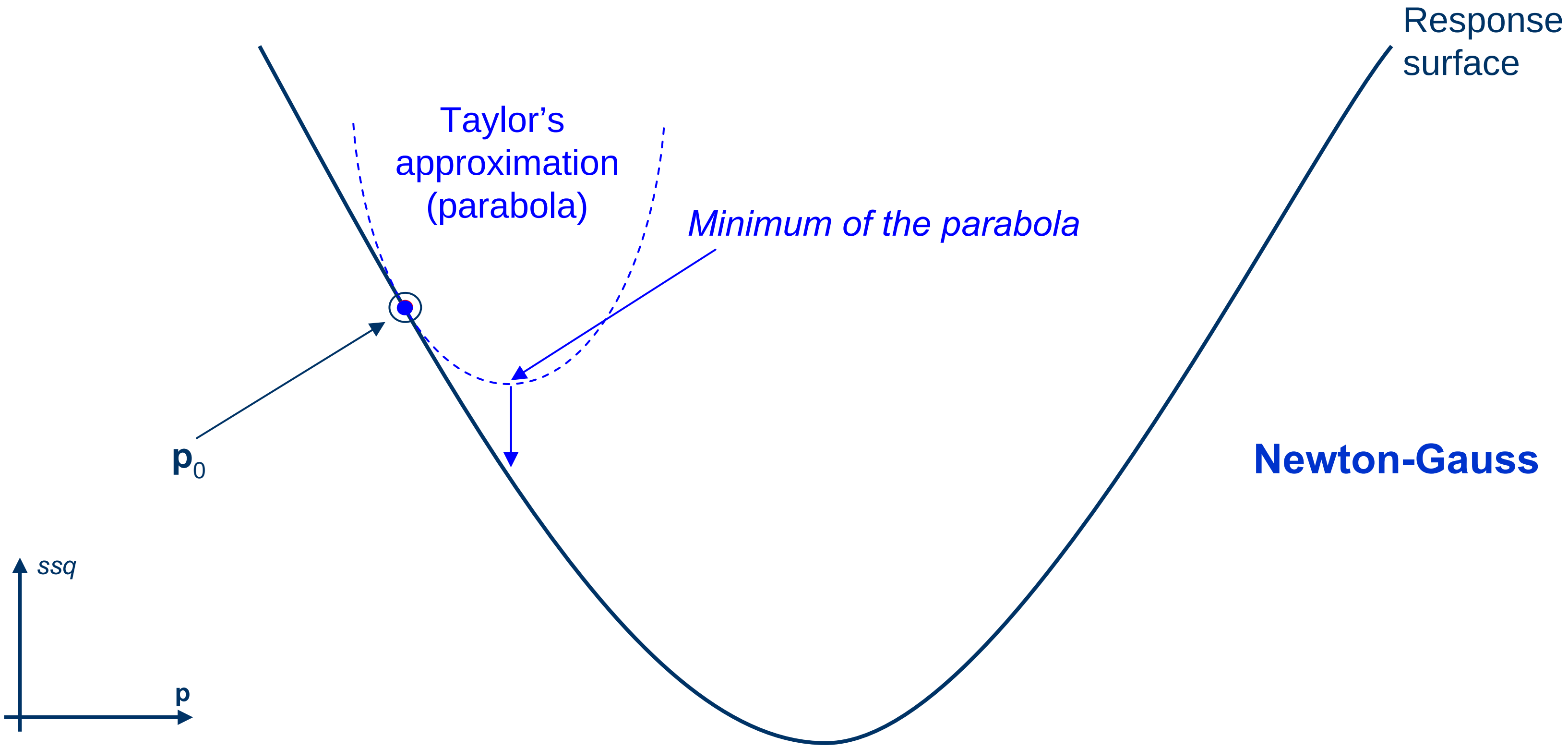
Newton-Gauss

Geometrical interpretation on the response surface



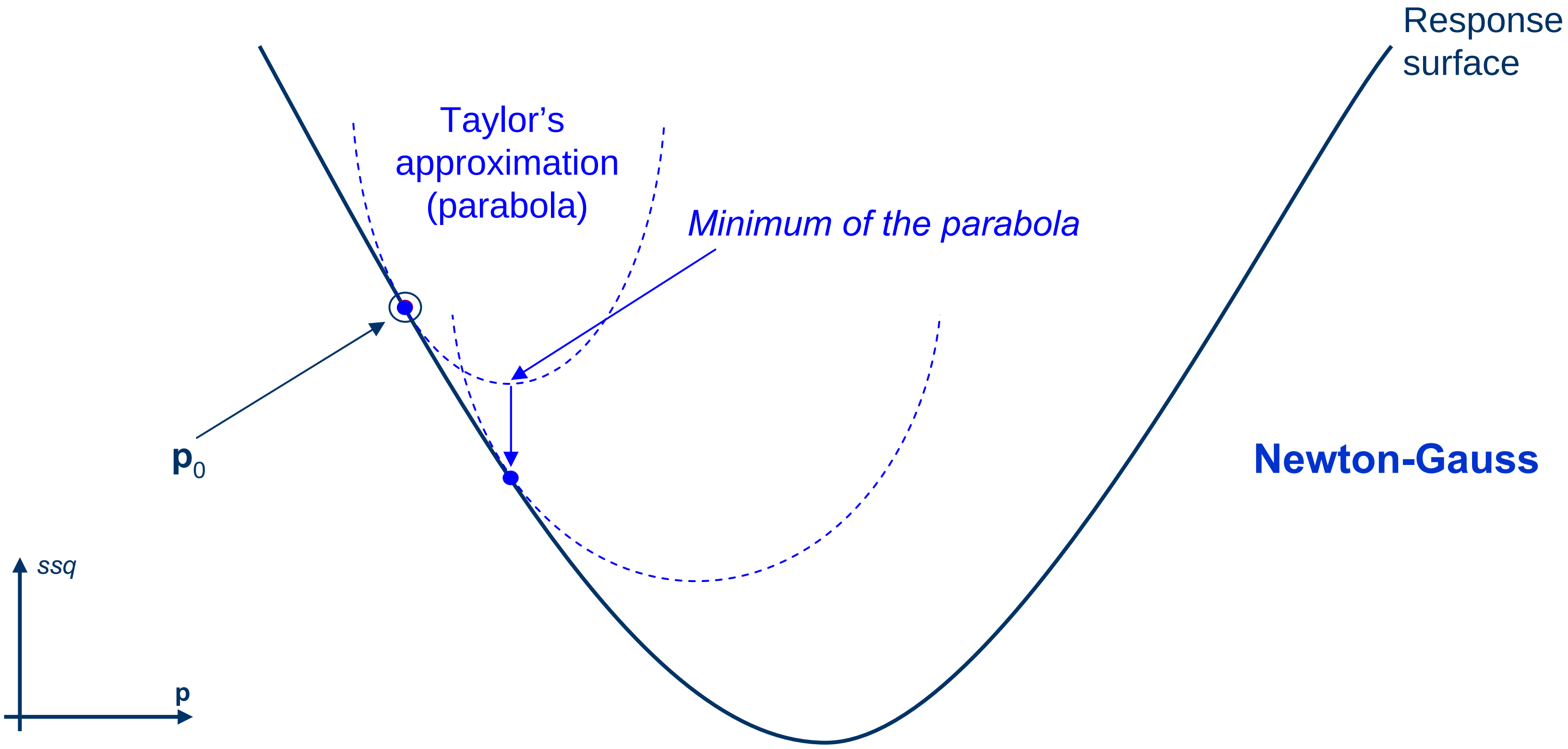
Newton-Gauss

Geometrical interpretation on the response surface



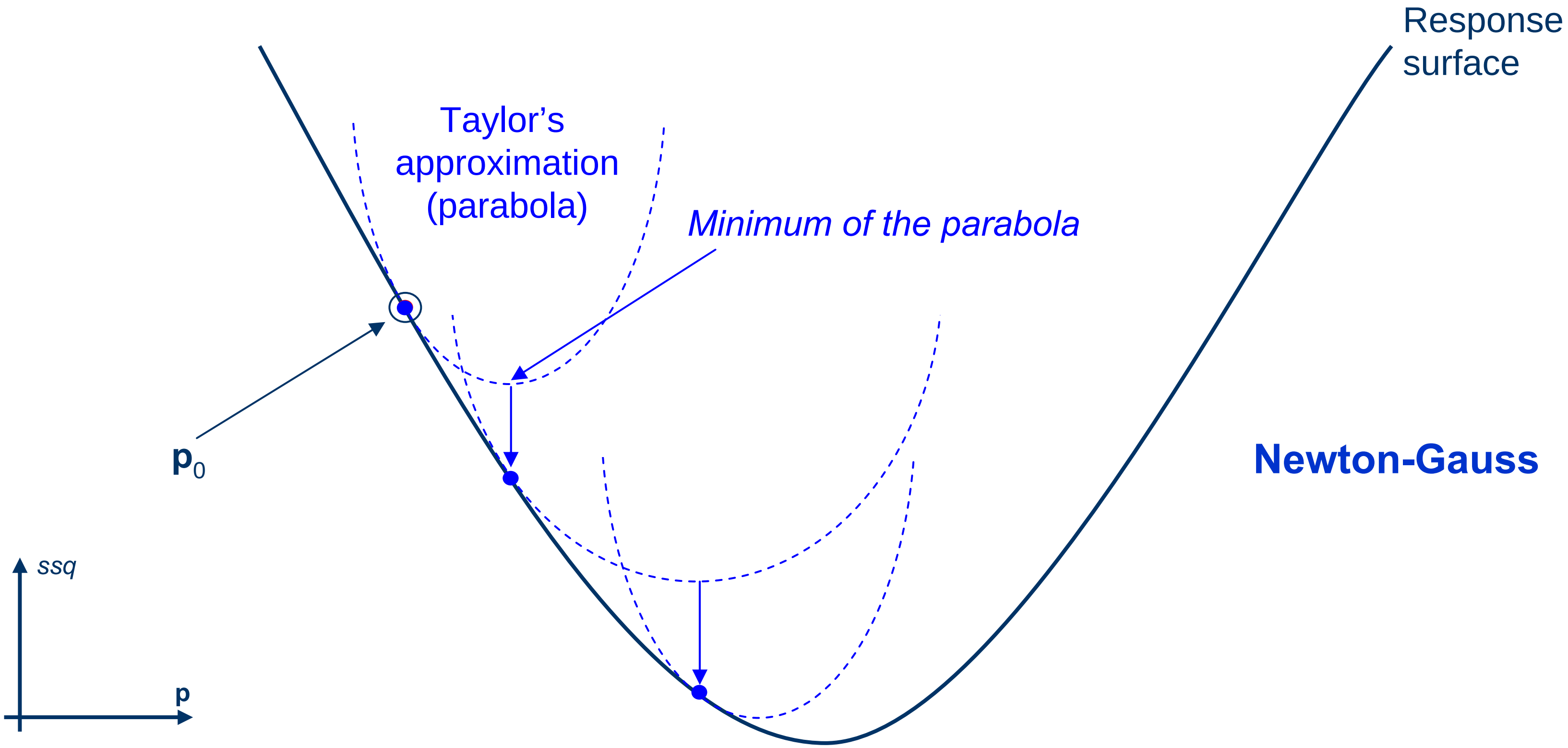
Newton-Gauss

Geometrical interpretation on the response surface



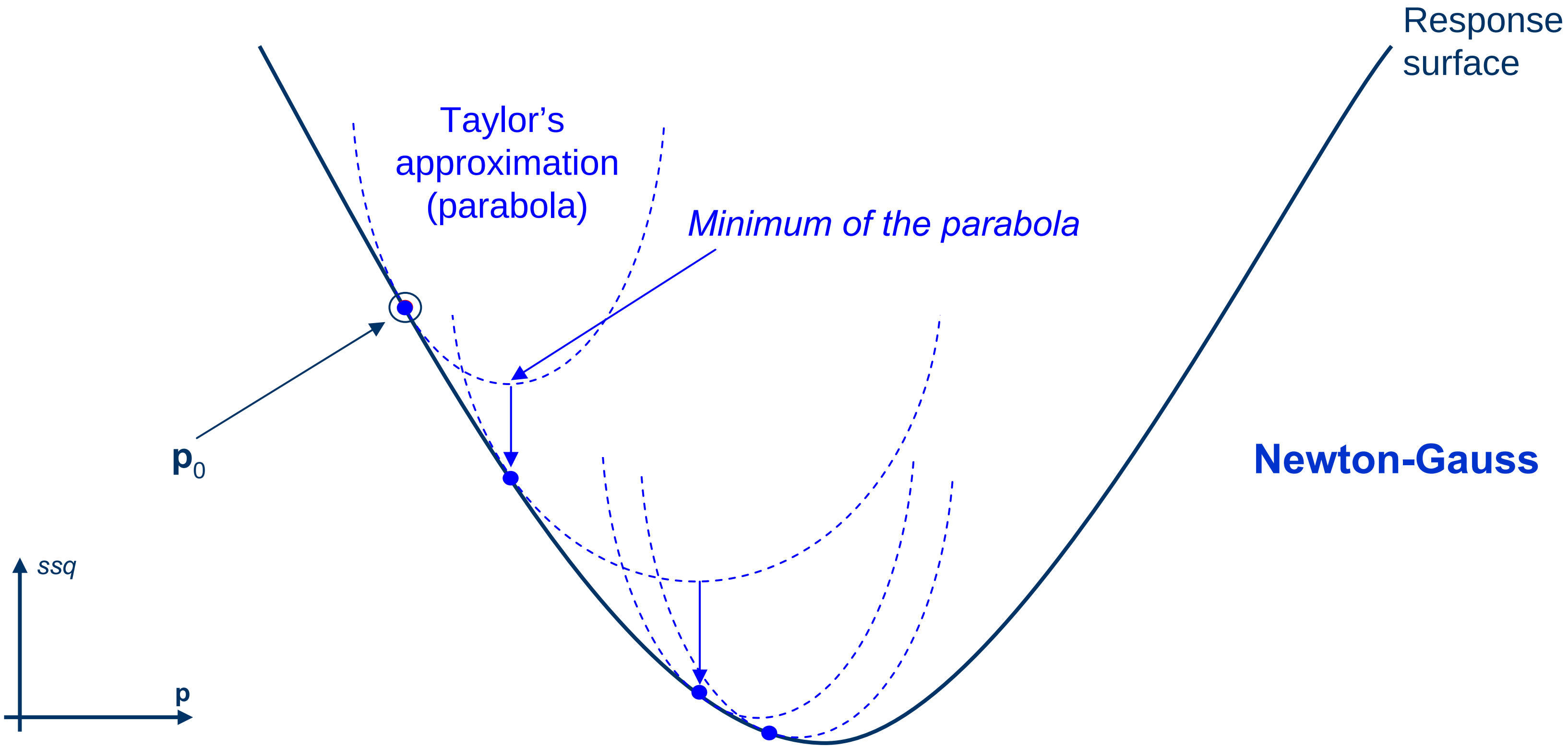
Newton-Gauss

Geometrical interpretation on the response surface



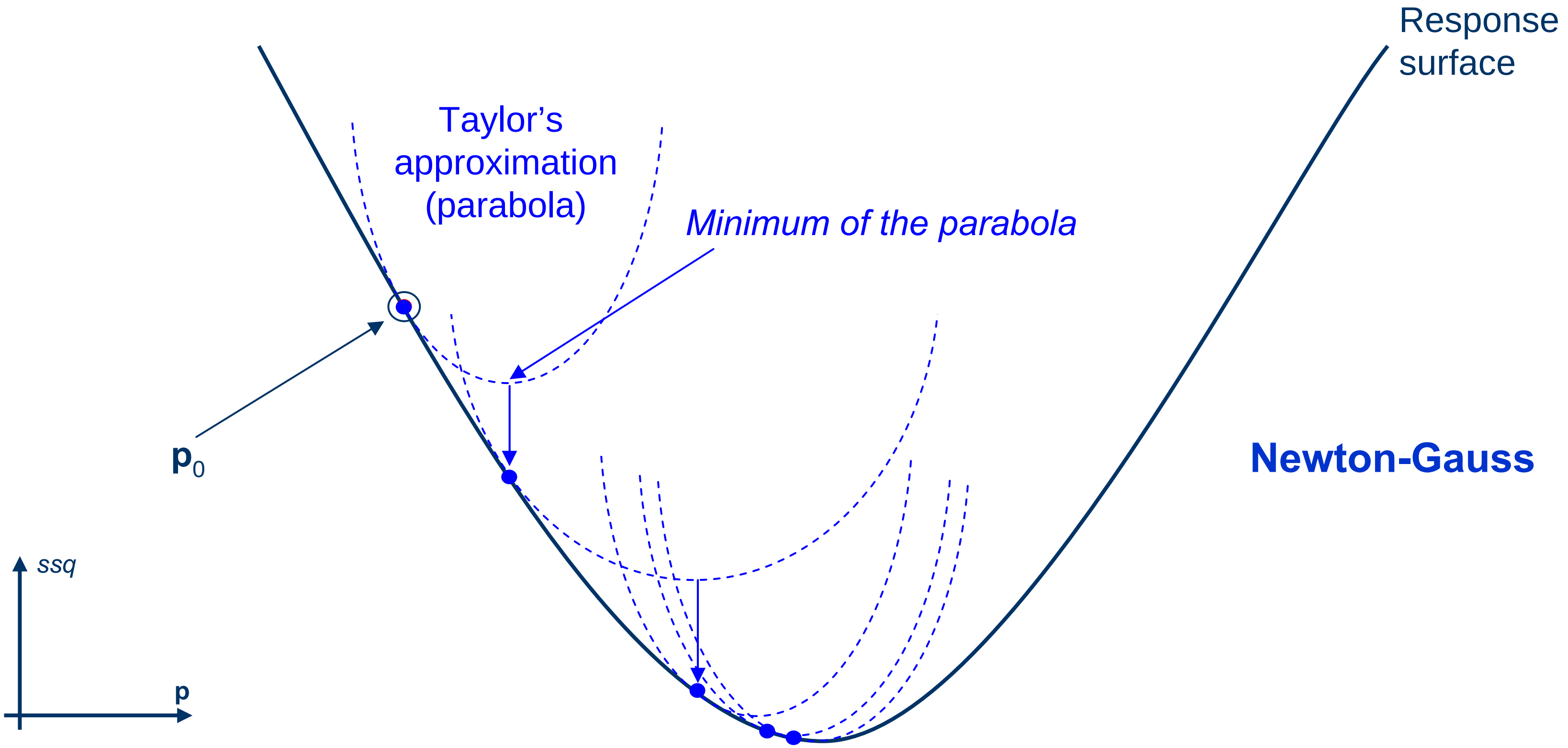
Newton-Gauss

Geometrical interpretation on the response surface



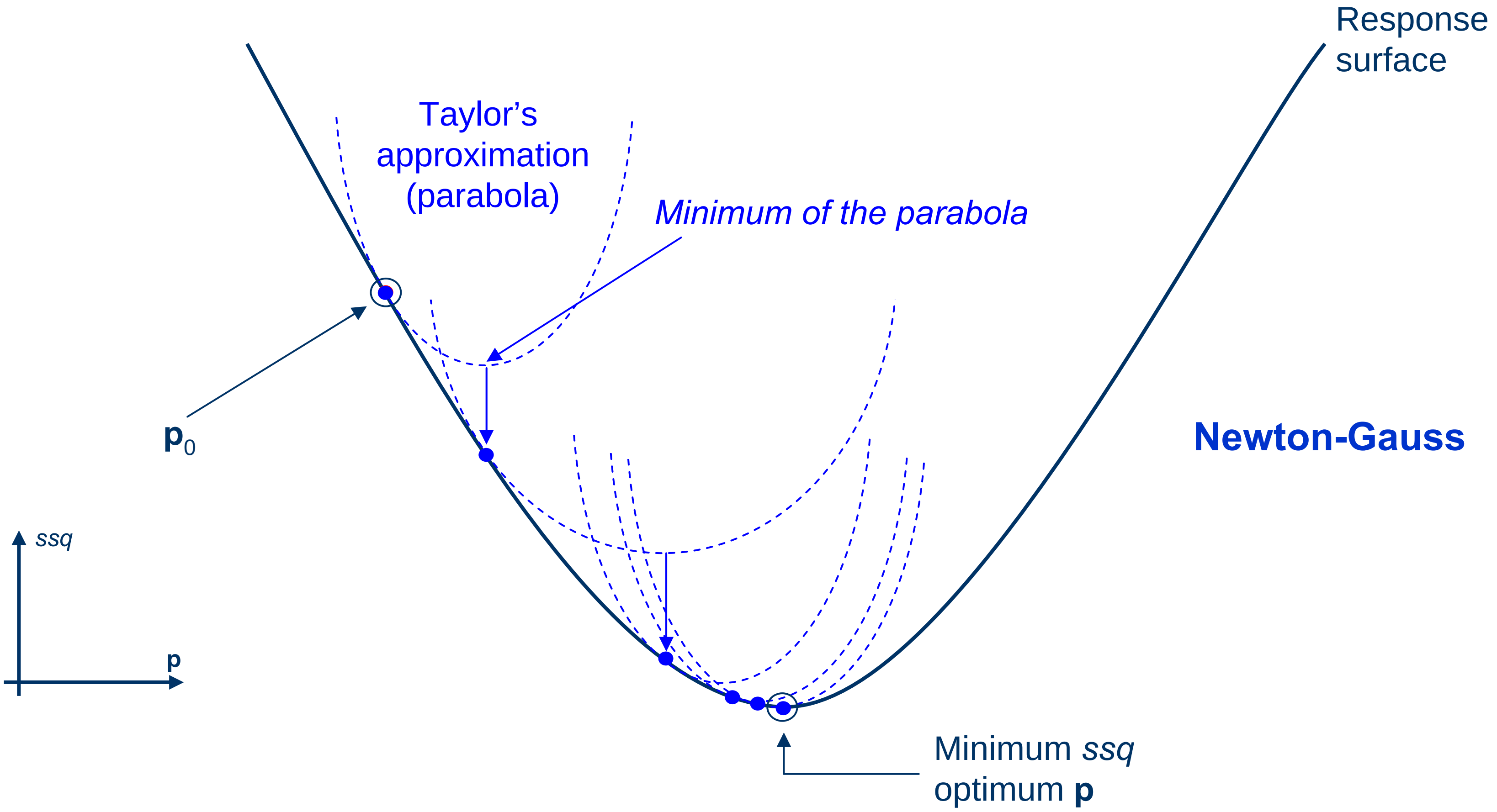
Newton-Gauss

Geometrical interpretation on the response surface



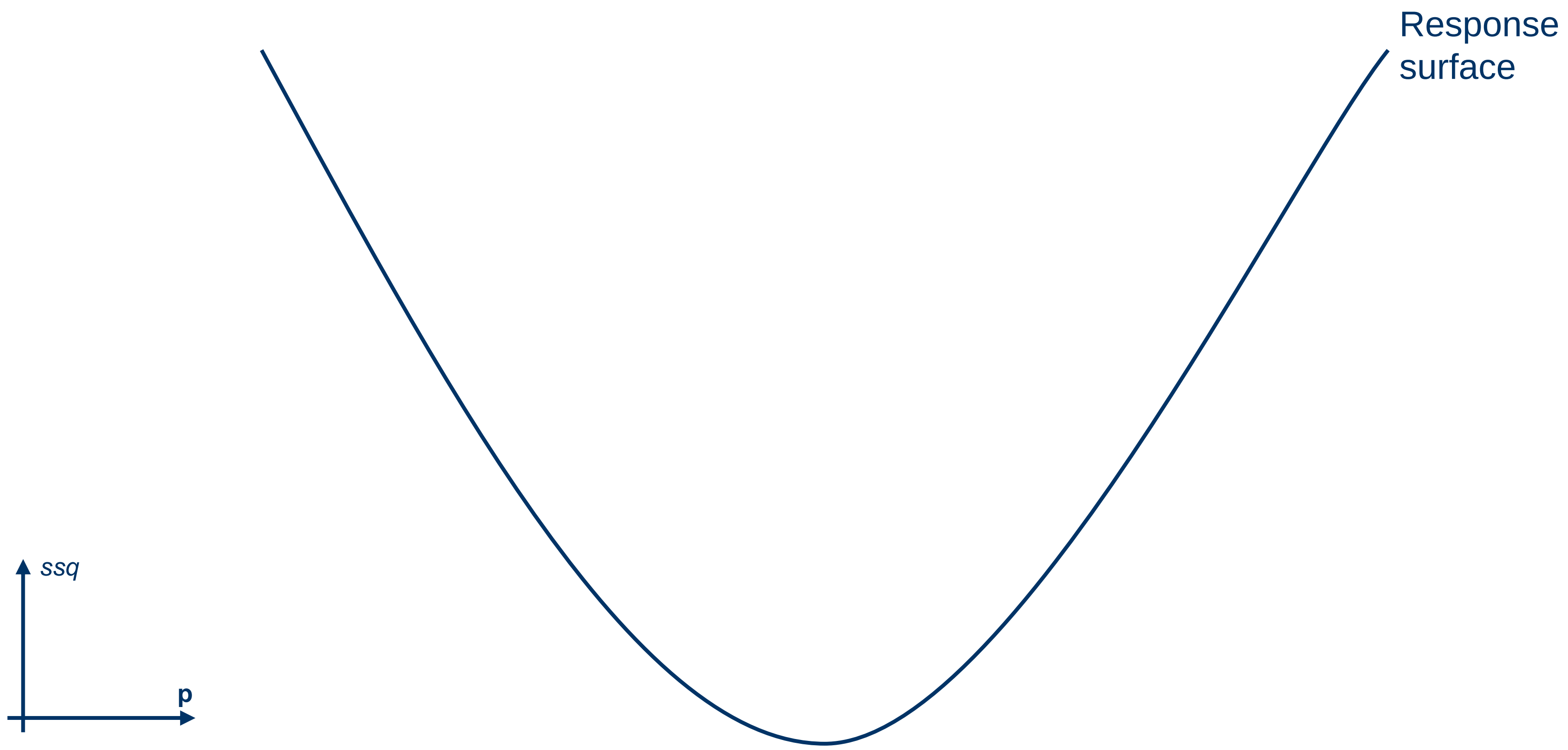
Newton-Gauss

Geometrical interpretation on the response surface

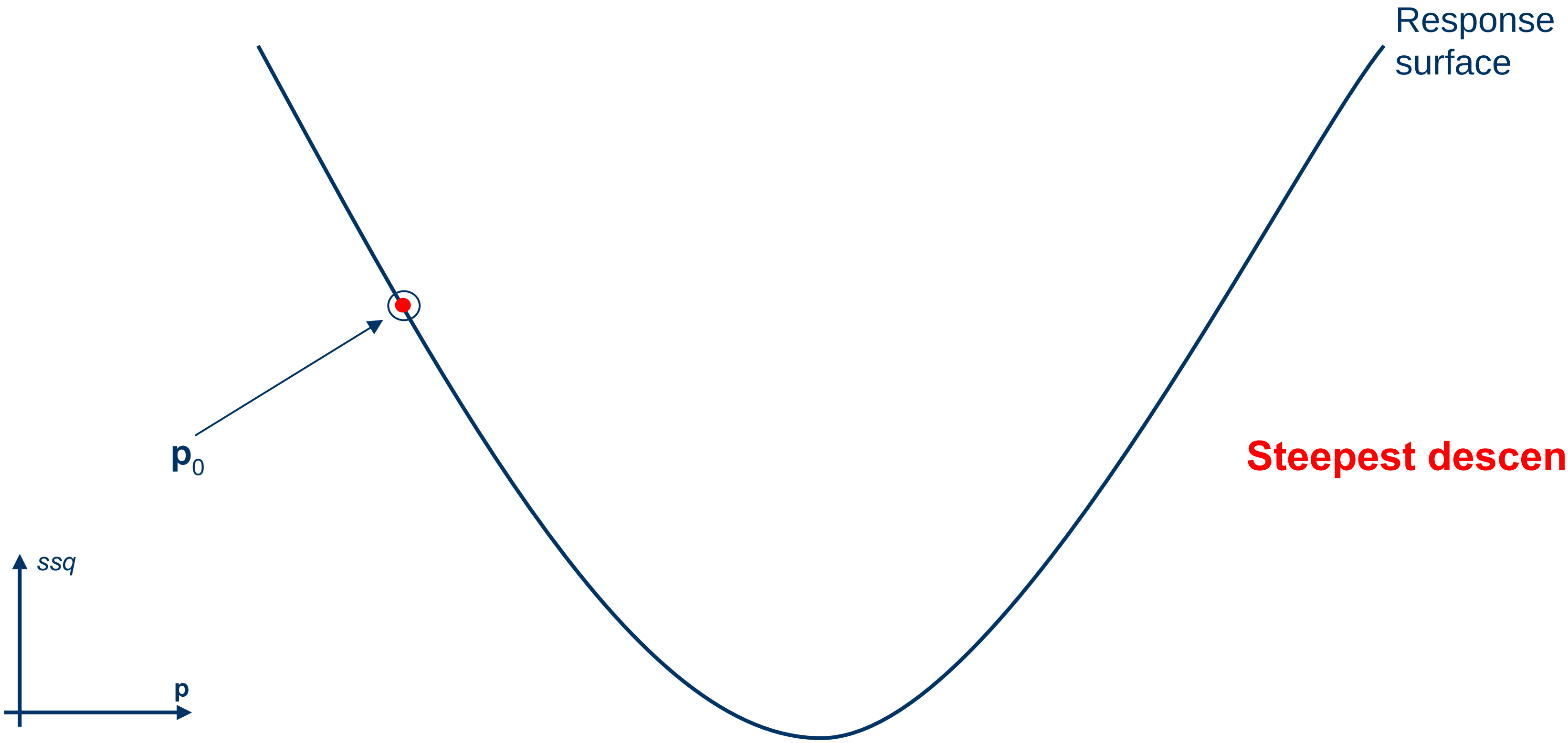

Response surface
Taylor's approximation (parabola)
Minimum of the parabola
p$_0$
ssq
p
Minimum *ssq* optimum **p**

Newton-Gauss

Geometrical interpretation on the response surface

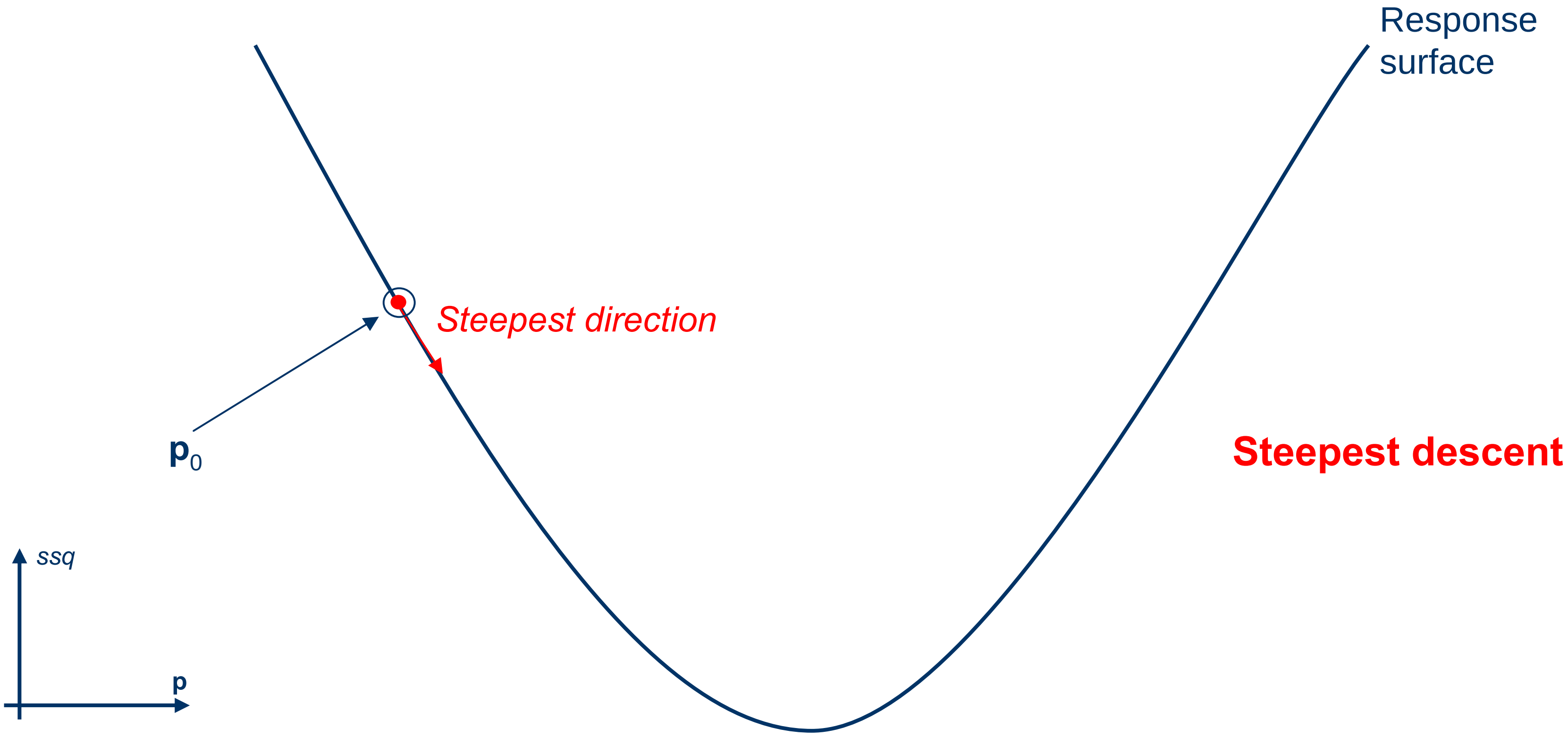


Geometrical interpretation on the response surface



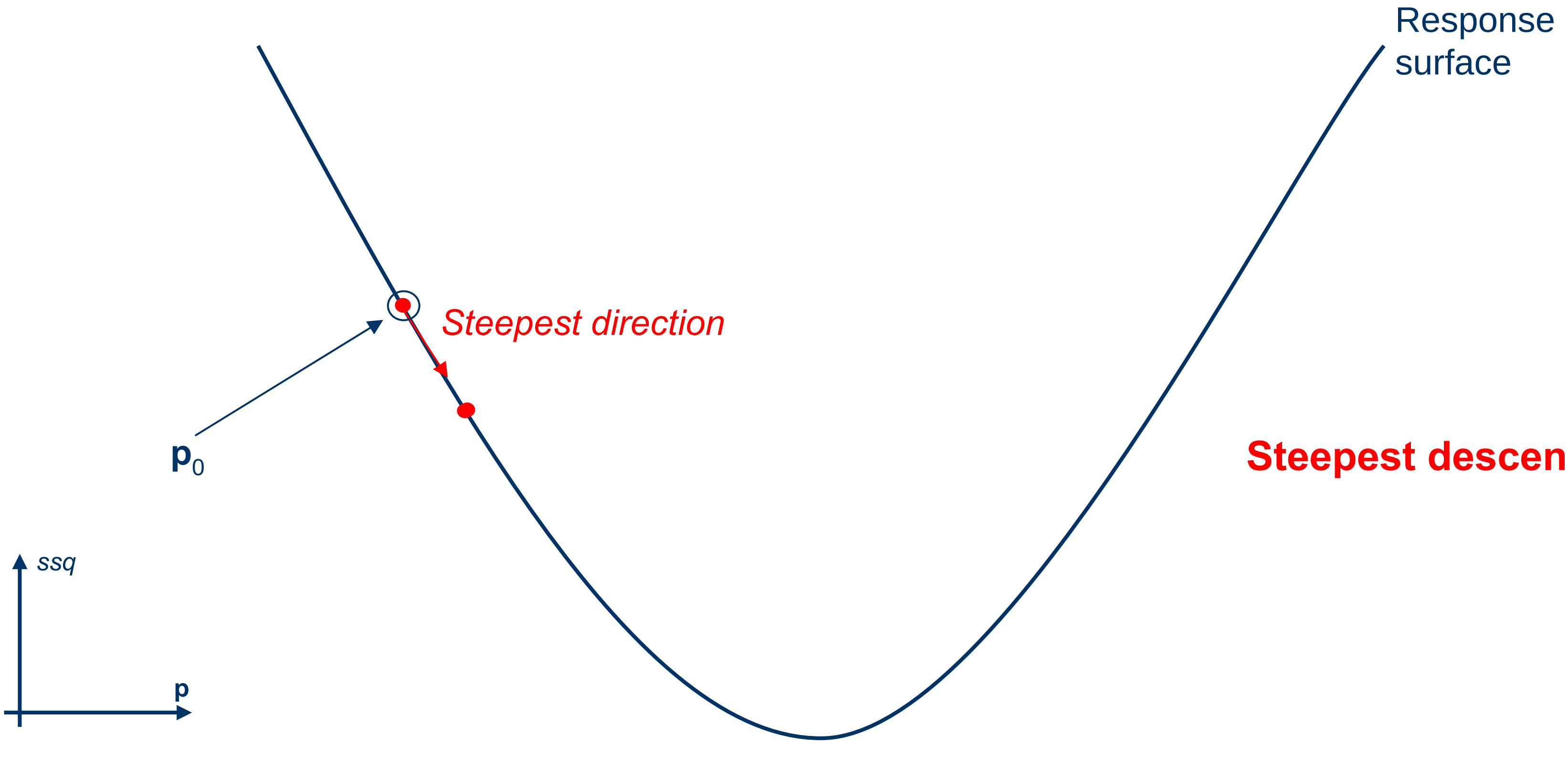
Steepest descent

Geometrical interpretation on the response surface



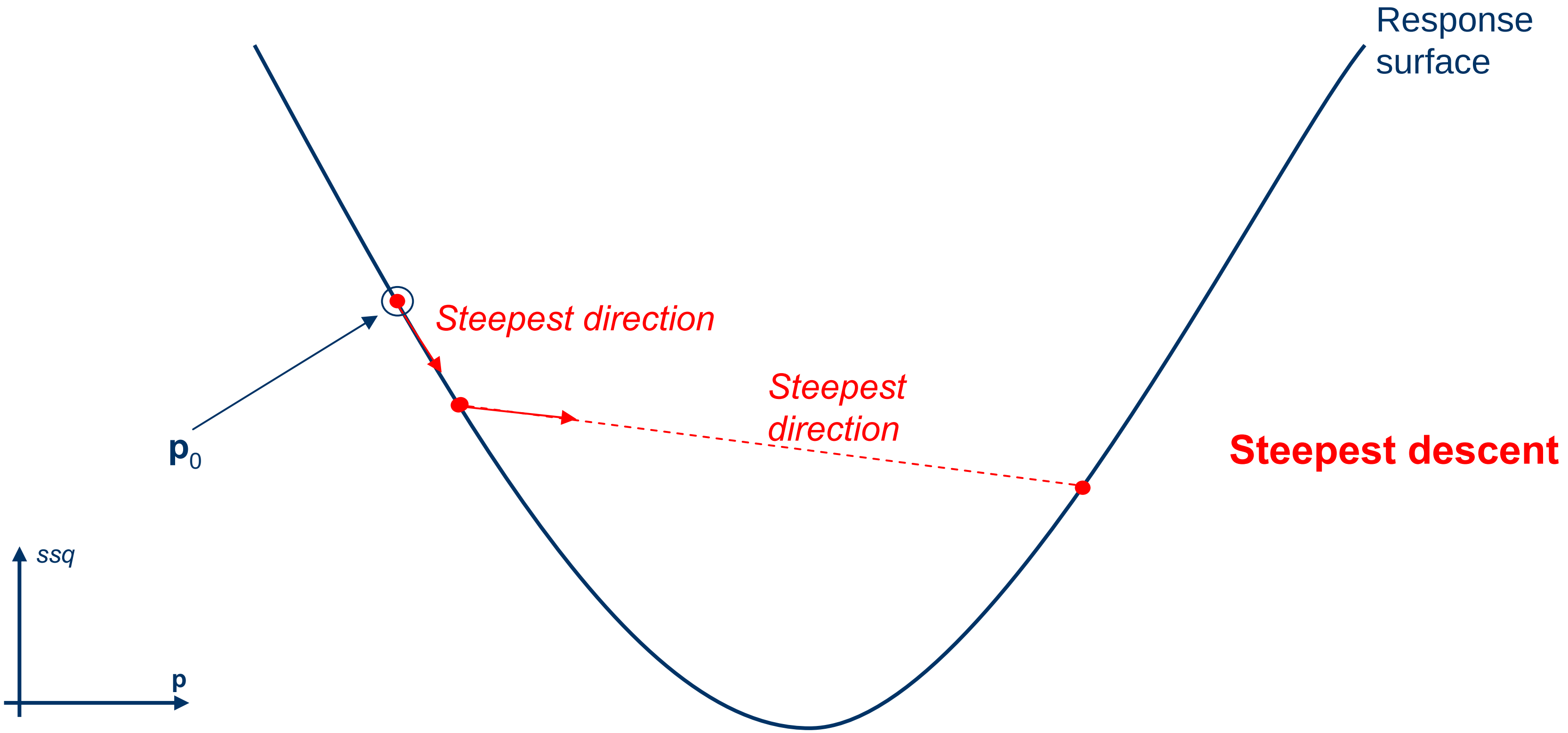
Steepest descent

Geometrical interpretation on the response surface



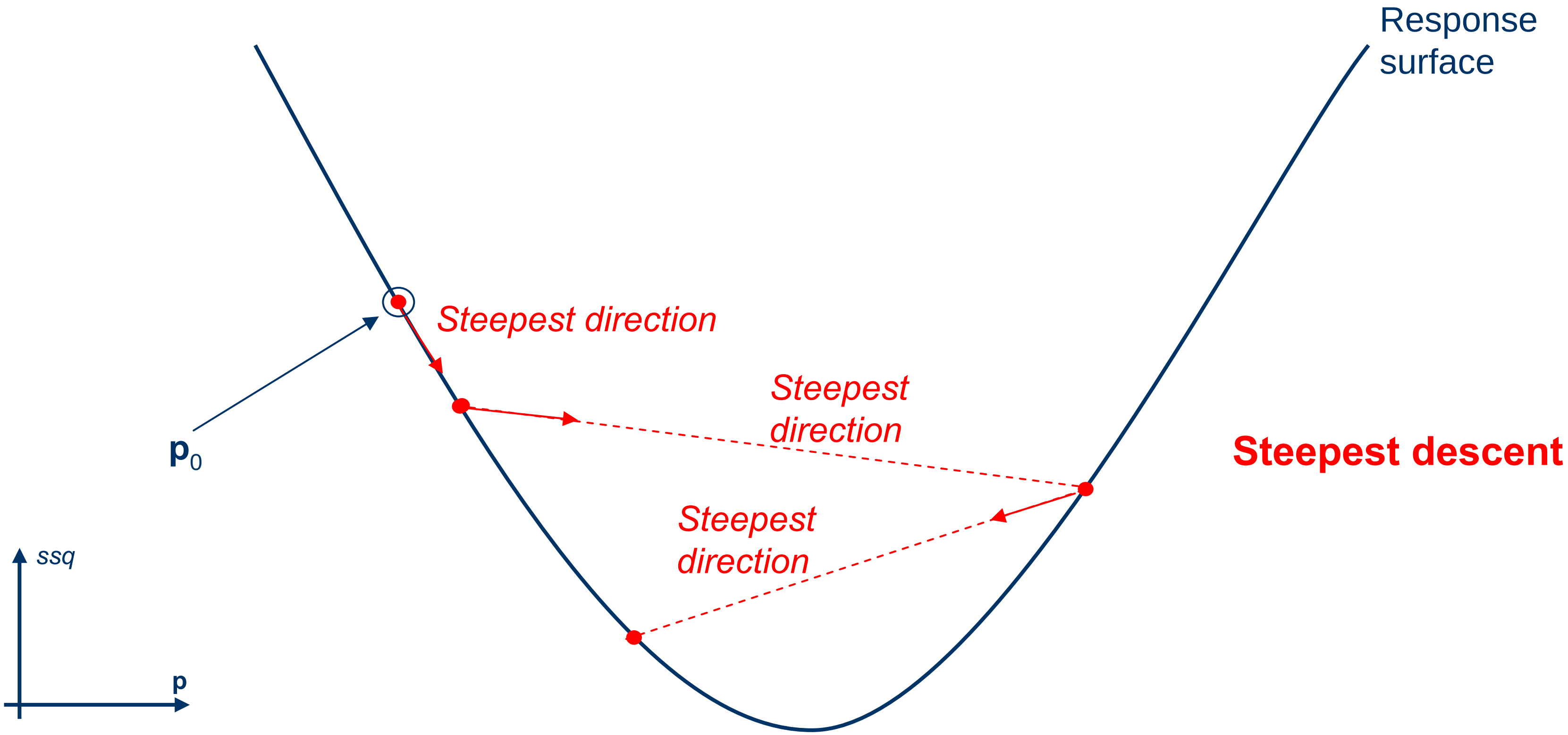
Steepest descent

Geometrical interpretation on the response surface



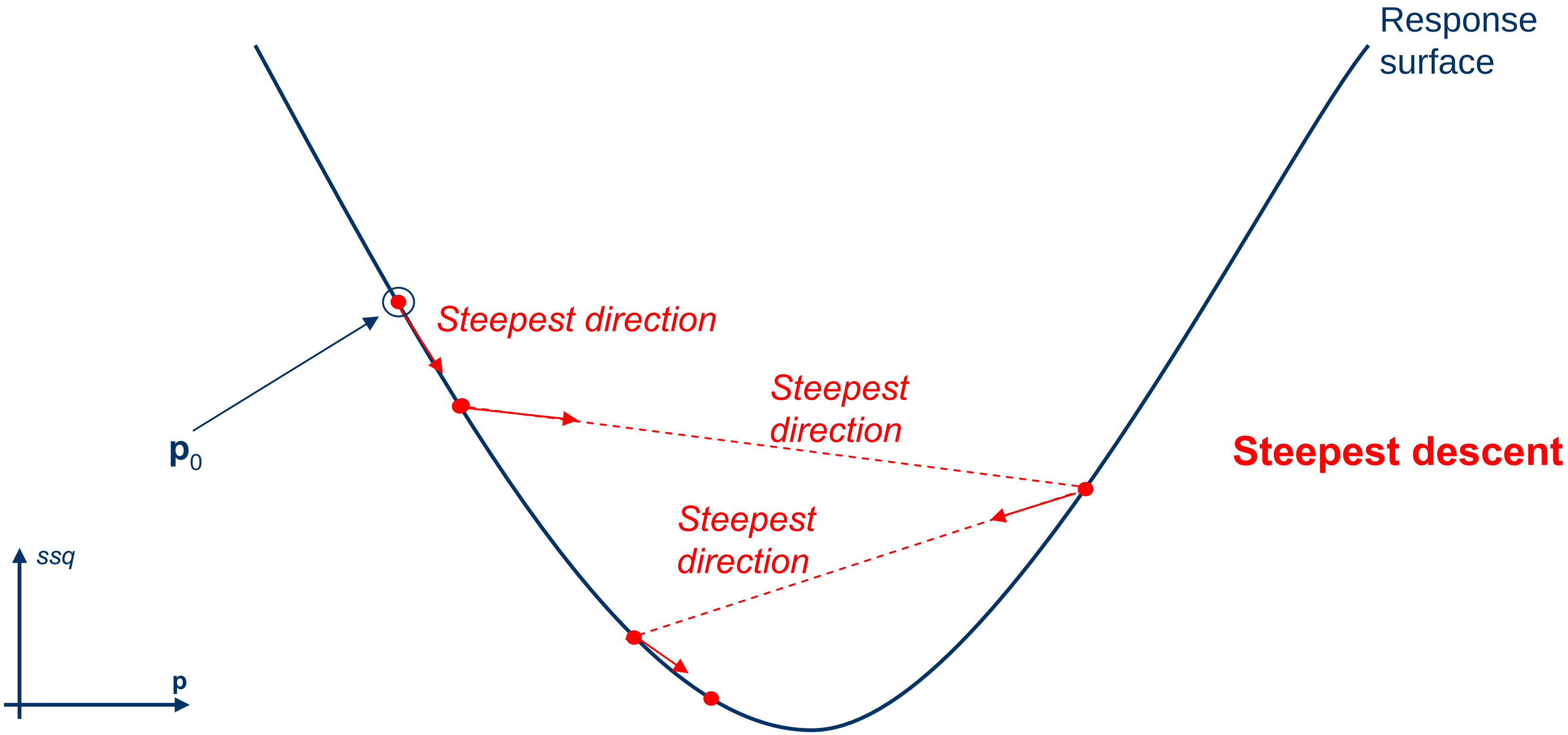
Steepest descent

Geometrical interpretation on the response surface

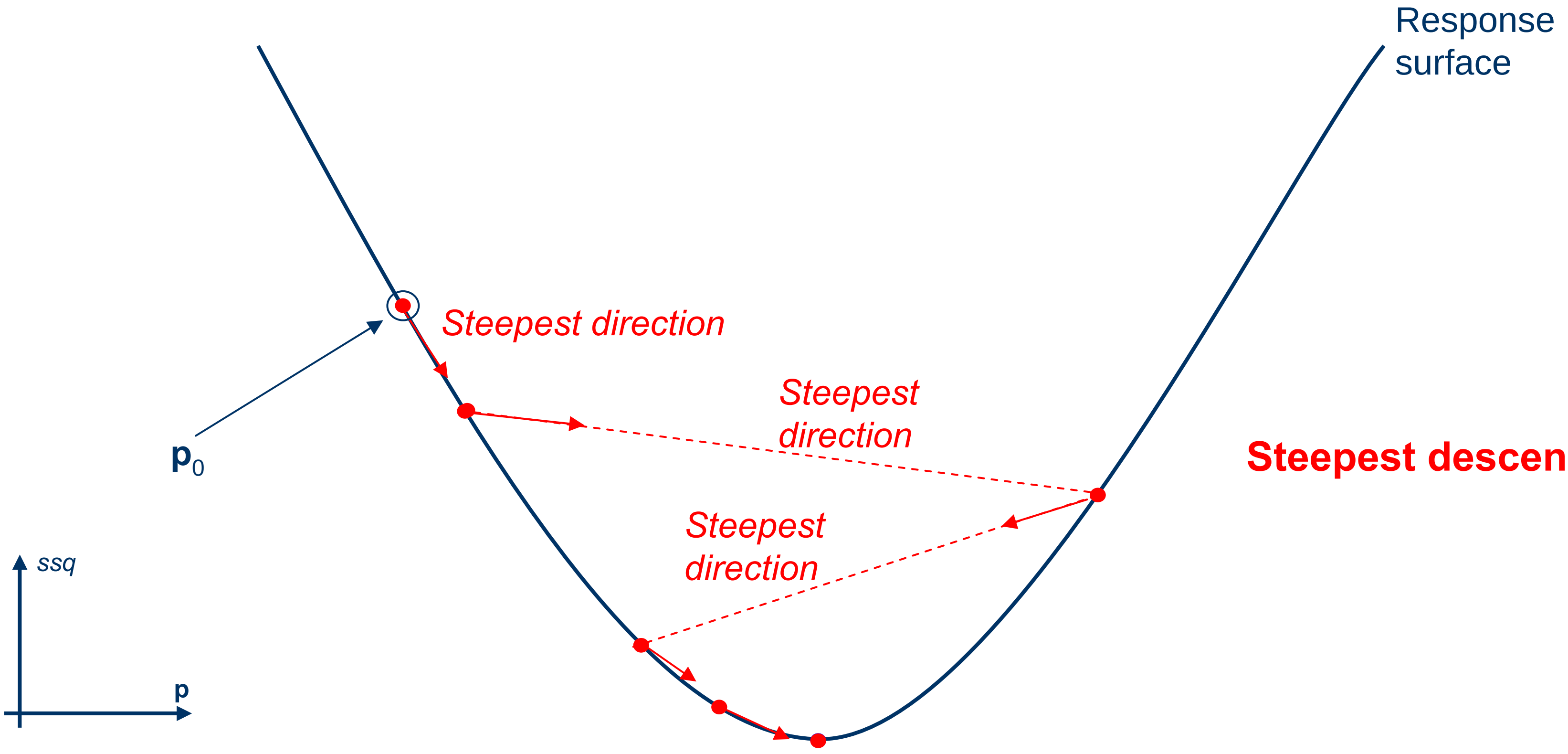


Steepest descent

Geometrical interpretation on the response surface

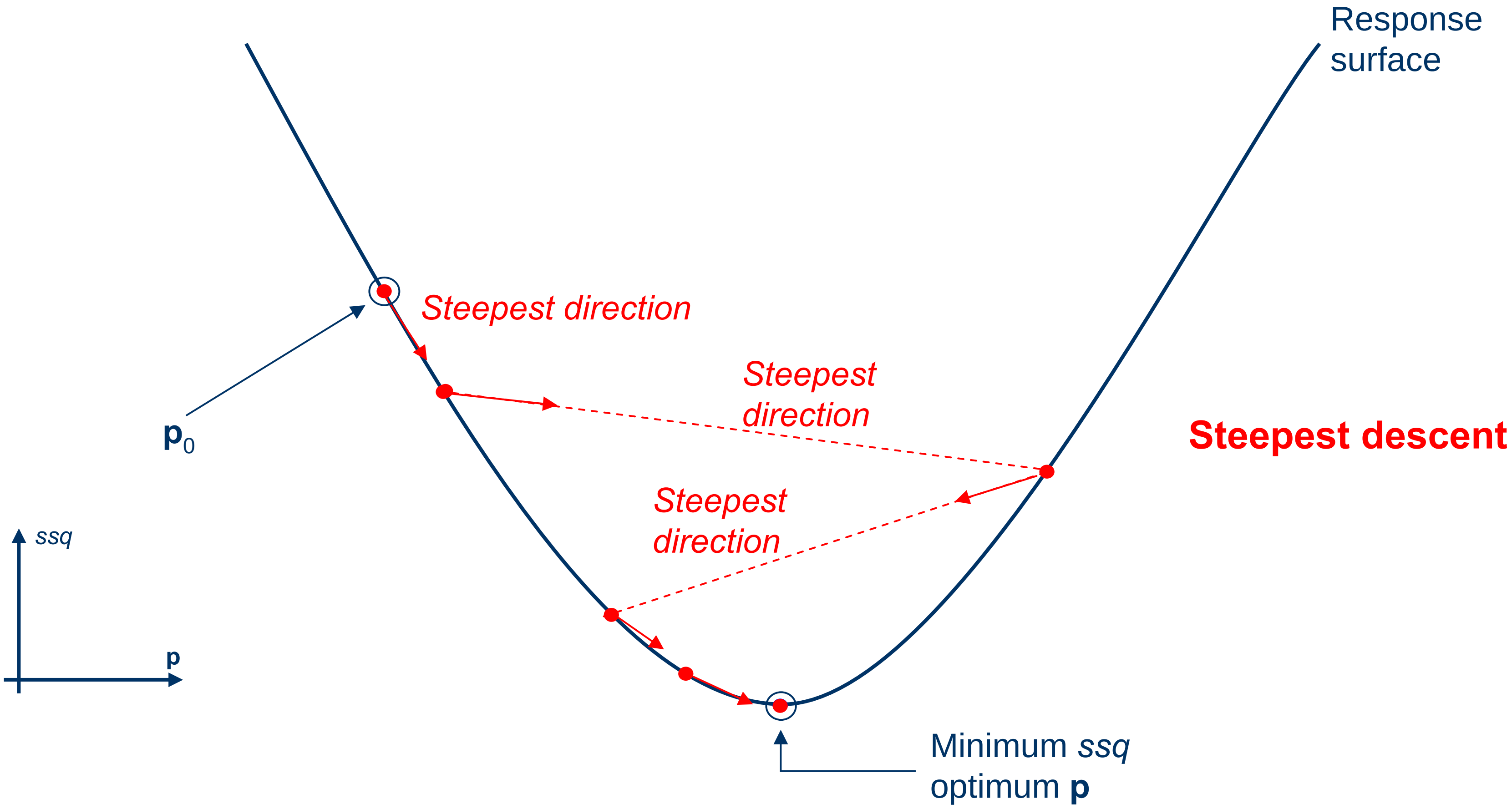


Geometrical interpretation on the response surface



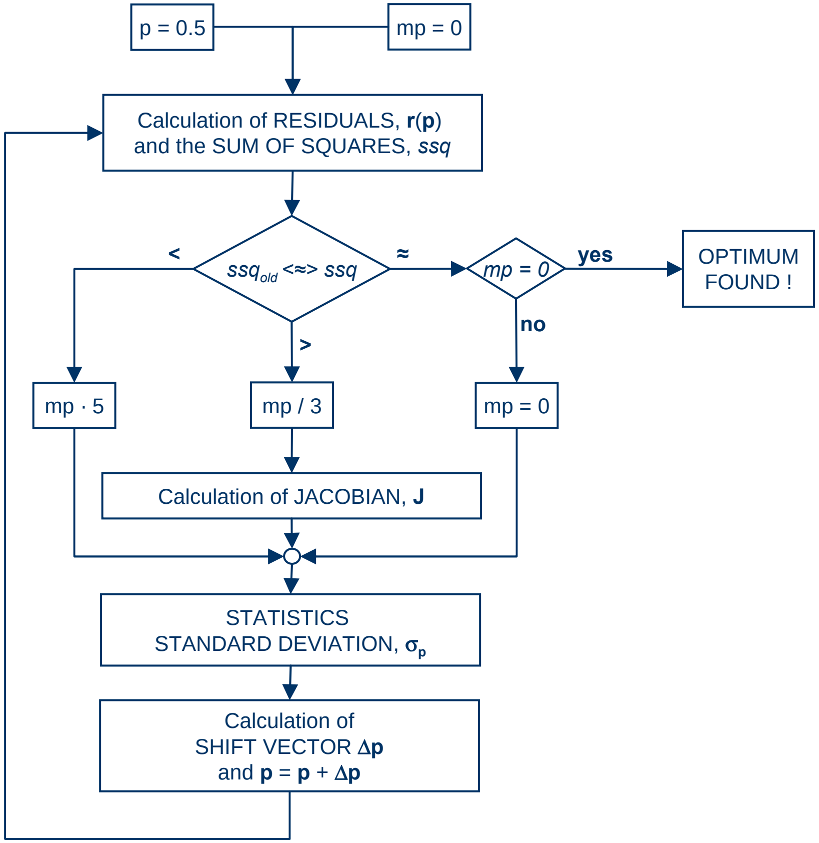
Steepest descent

Geometrical interpretation on the response surface



Steepest descent

The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

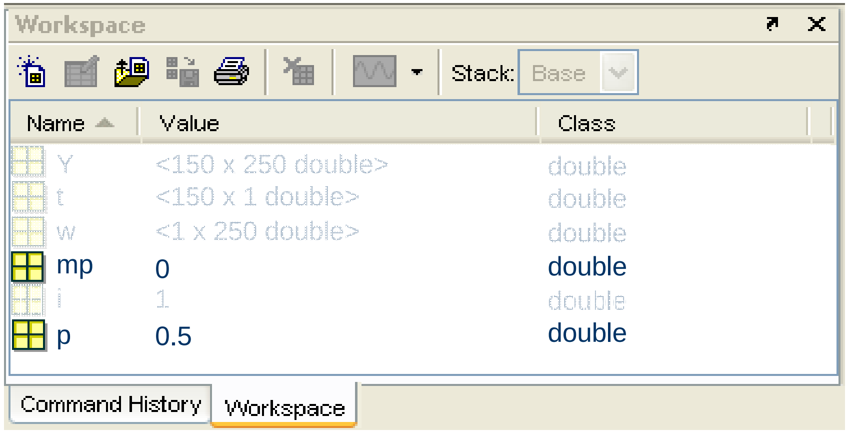
mp = 0

Calculation of JACOBIAN, **J**

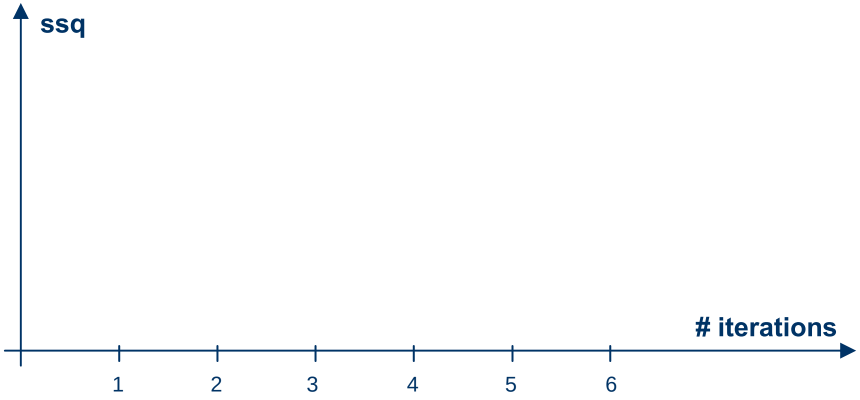
STATISTICS
STANDARD DEVIATION, σ_p

Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

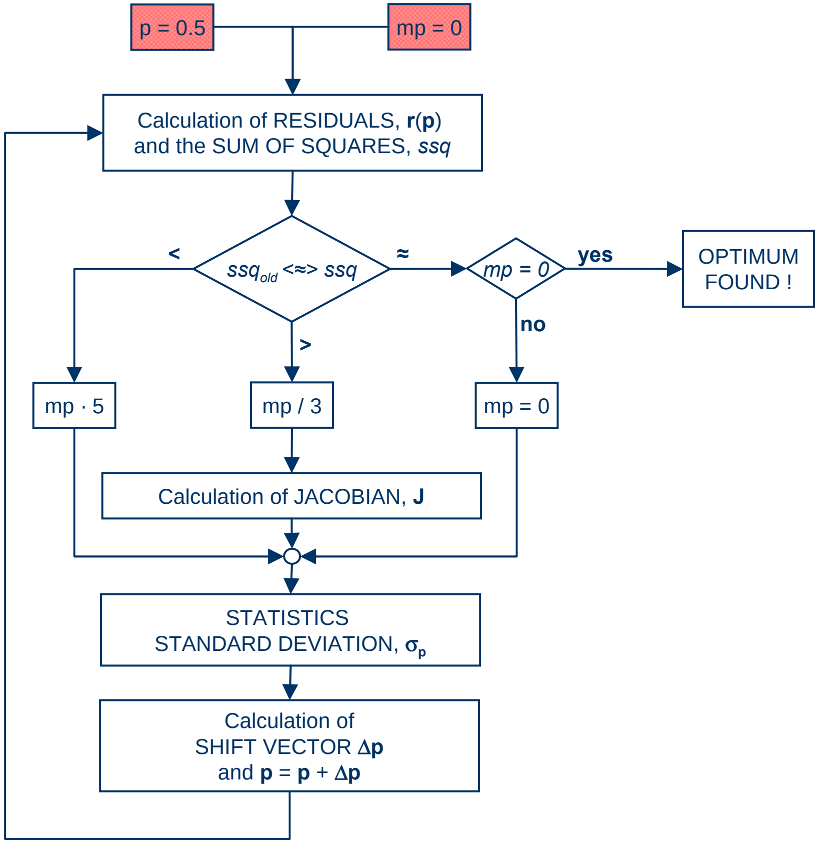
Current method :



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.5	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

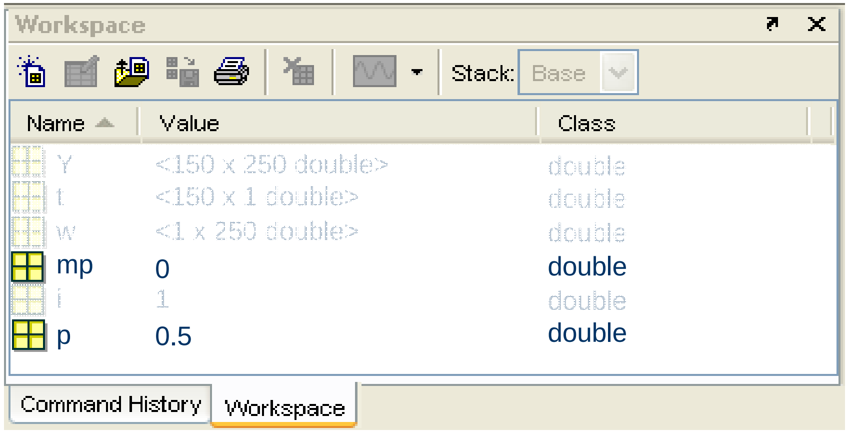
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

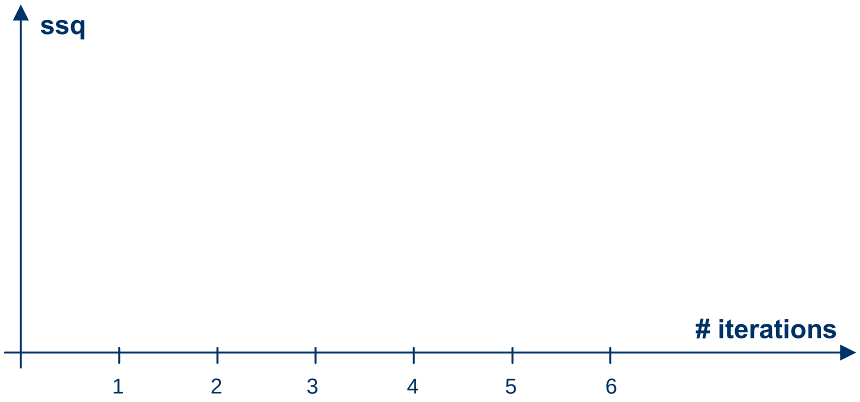
Calculation of SHIFT VECTOR **Δp** and **p** = **p** + **Δp**

Current method :

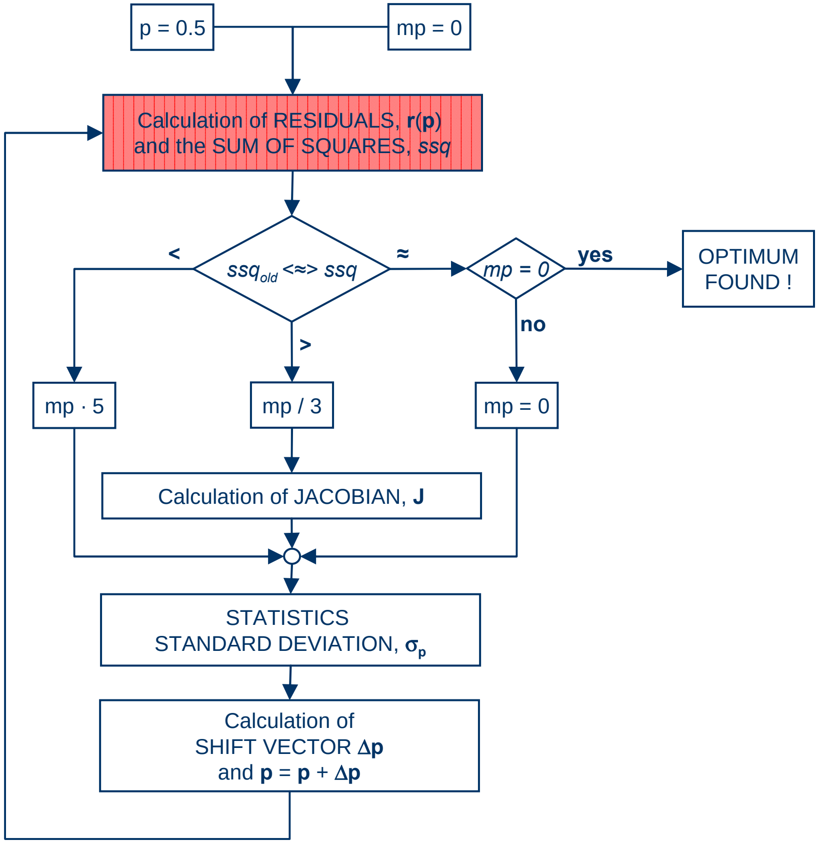
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.5	double

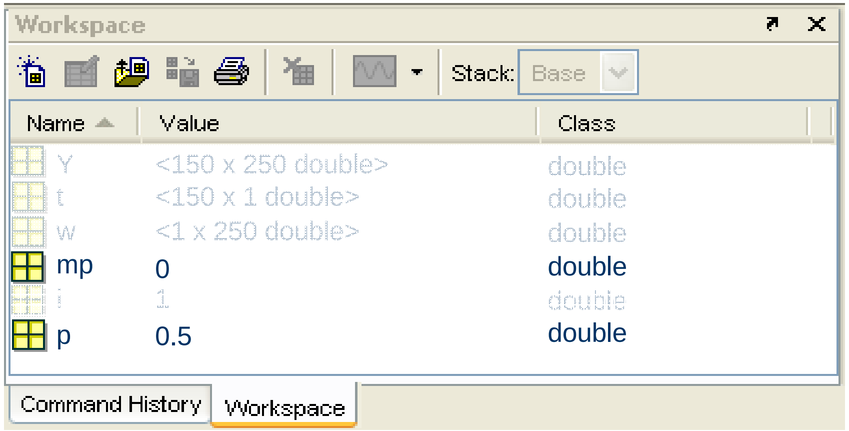


The NG/LM algorithm

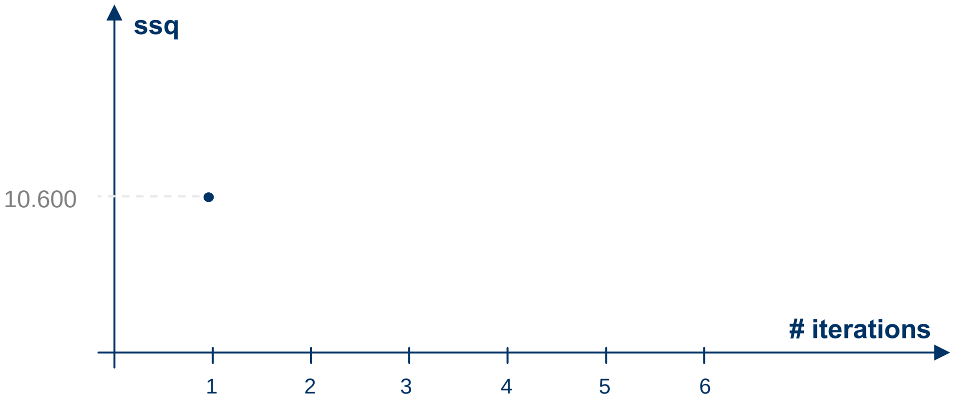

p = 0.5
mp = 0
Calculation of RESIDUALS, **r(p)**
and the SUM OF SQUARES, *ssq*
<
ssq_{old} <≈> *ssq*
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS
STANDARD DEVIATION, σ_p
Calculation of
SHIFT VECTOR **Δp**
and **p** = **p** + **Δp**

Current method :

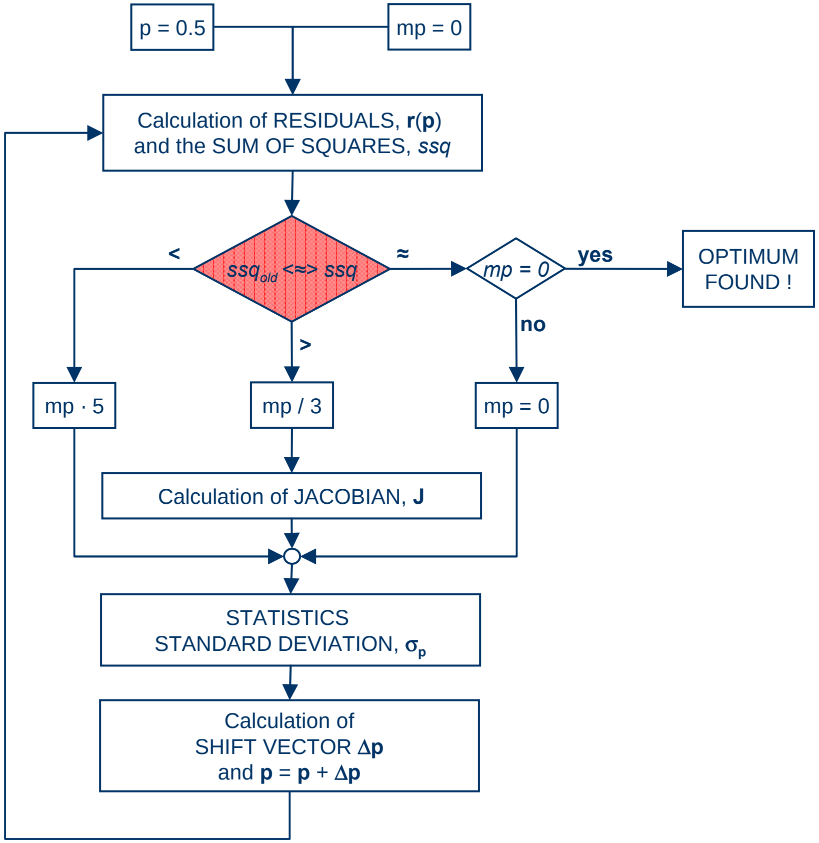
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.5	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> ssq

<

≈

>

$mp = 0$

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

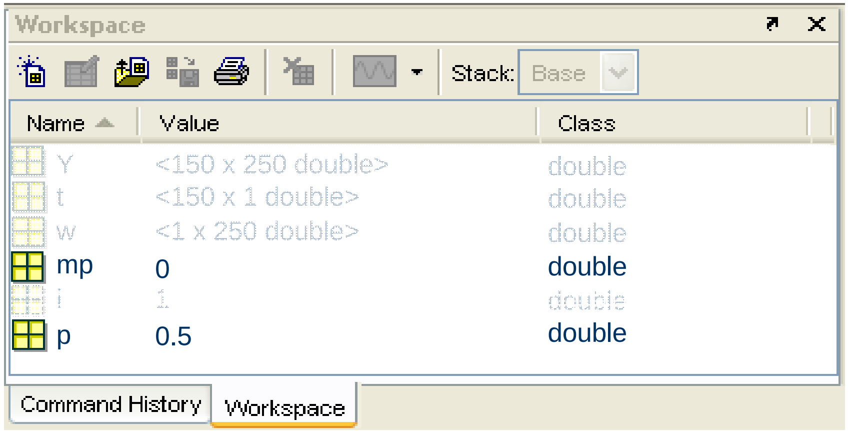
Calculation of JACOBIAN, **J**

STATISTICS
STANDARD DEVIATION, σ_p

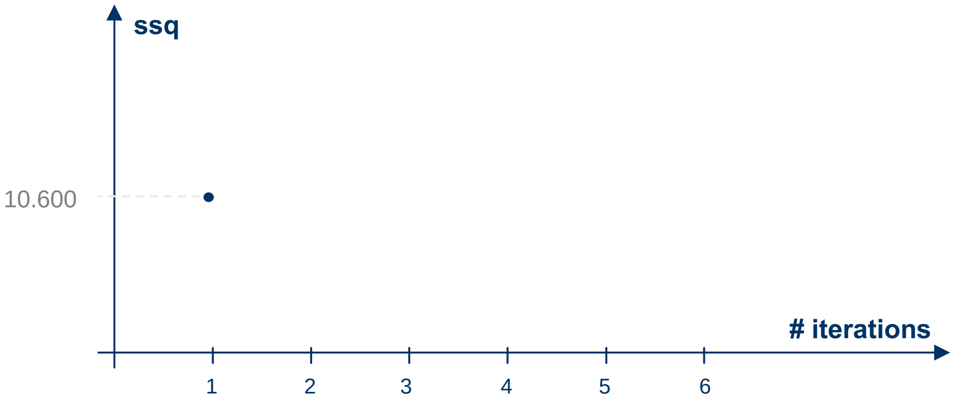
Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

Current method :

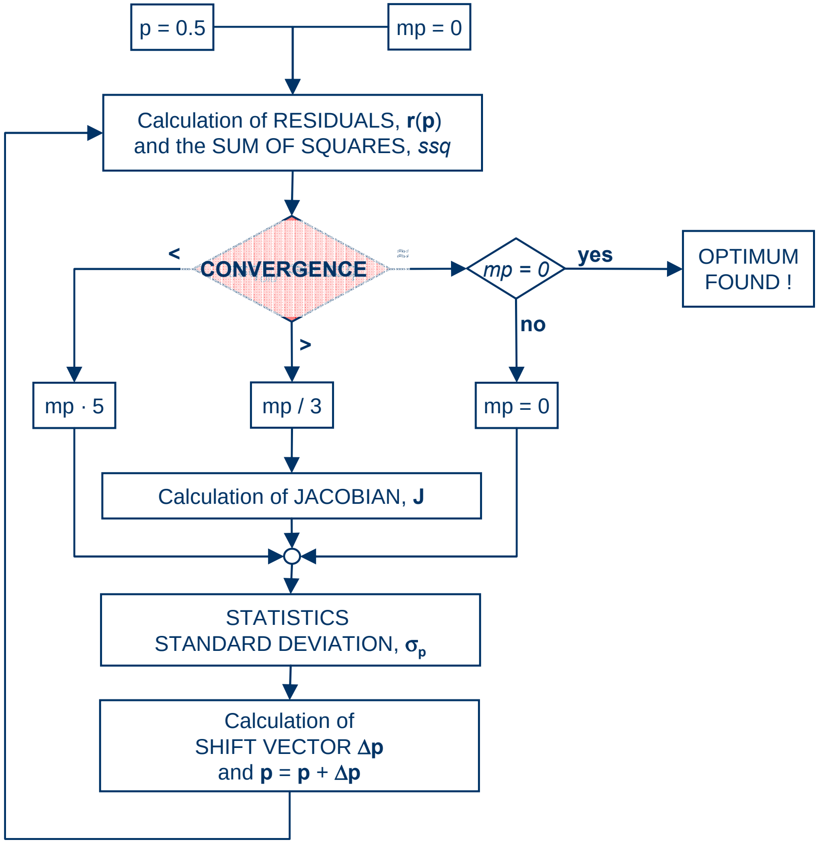
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.5	double



The NG/LM algorithm


p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

CONVERGENCE

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

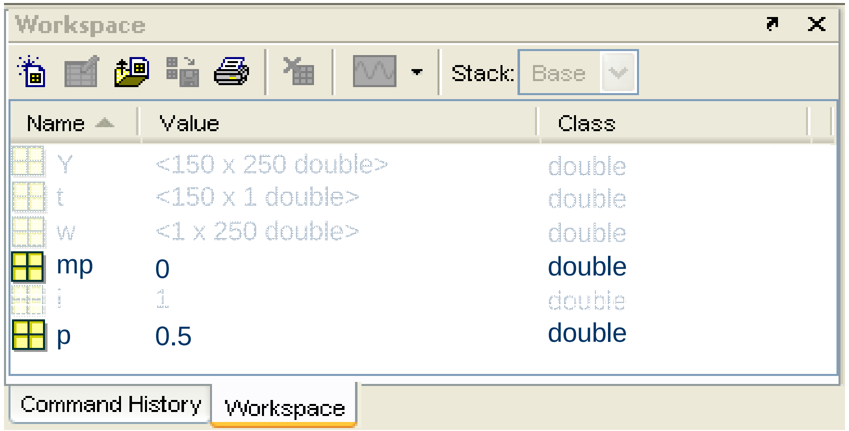
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

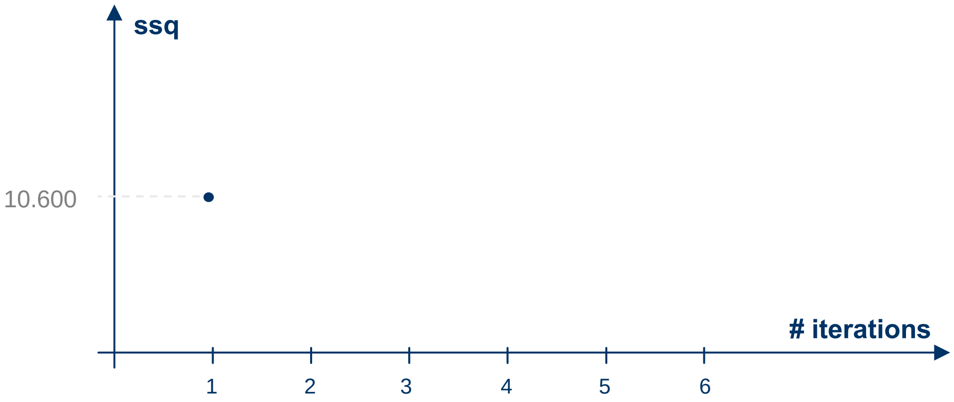
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

<u>Current method :</u>

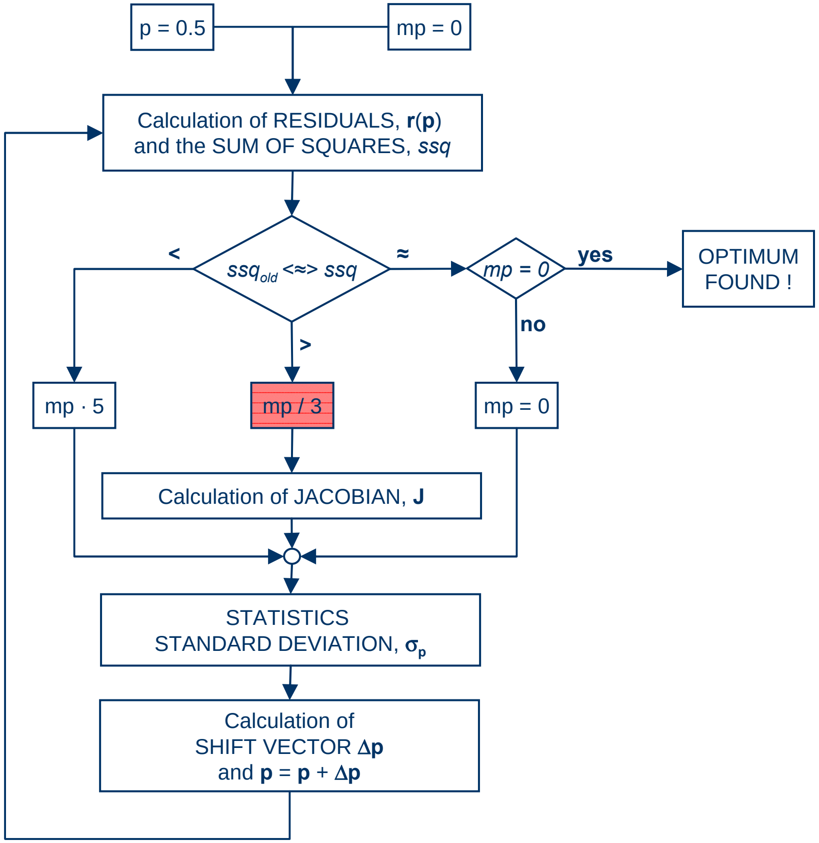
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.5	double



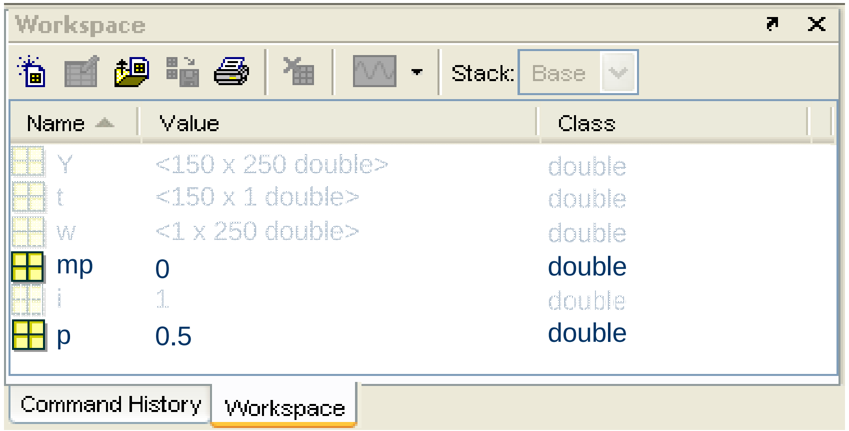
The NG/LM algorithm



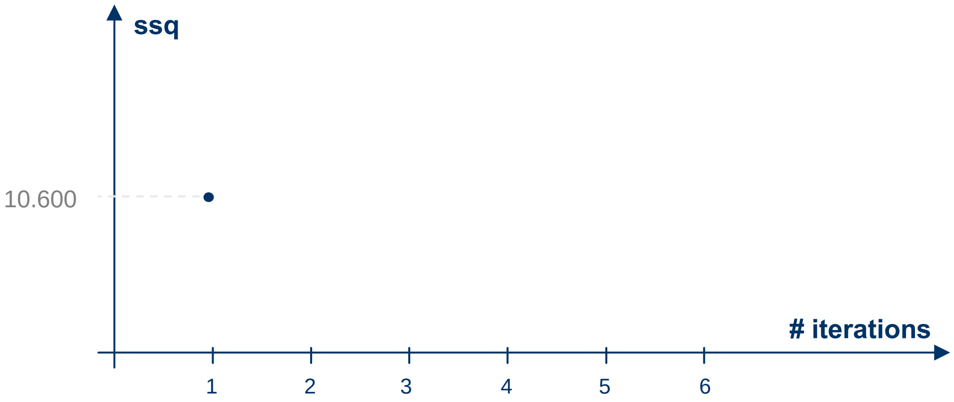
p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
ssq_{old} <≈> *ssq*
<
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

Current method :

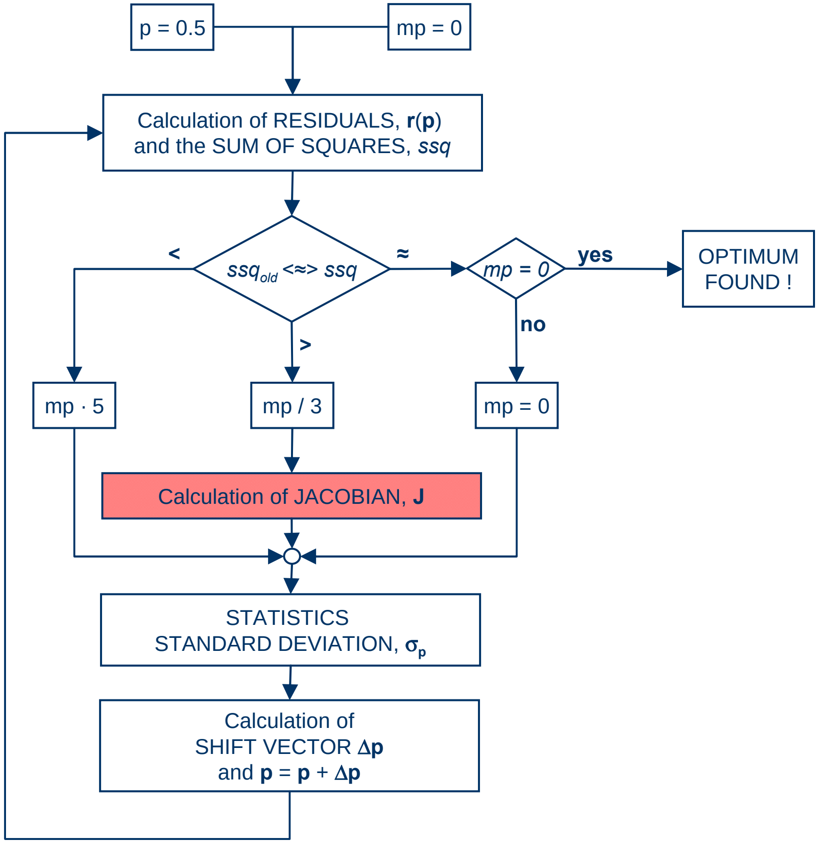
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.5	double

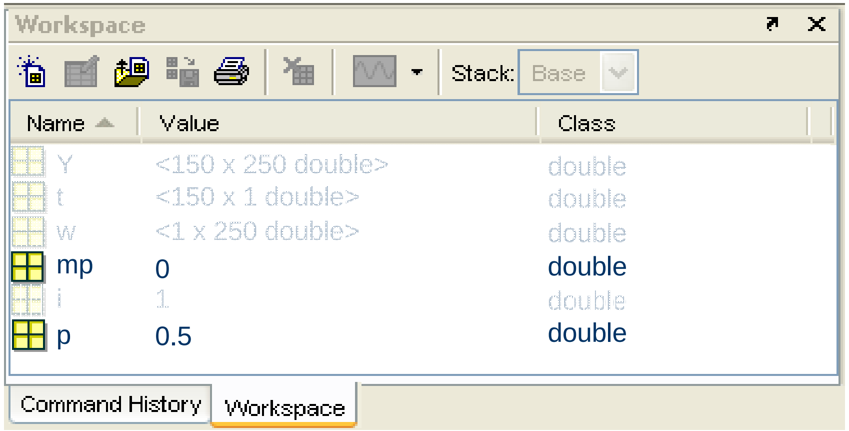


The NG/LM algorithm


p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
<
ssq_{old} <≈> ssq
≈
$mp = 0$
yes
OPTIMUM FOUND !
>
no
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

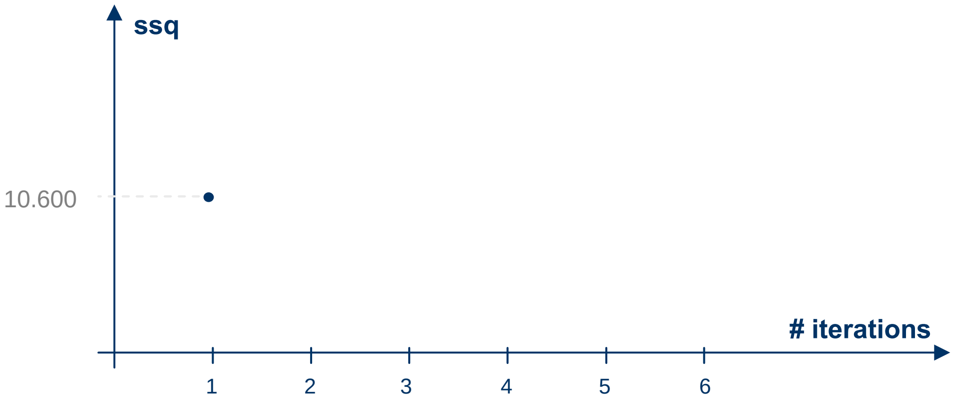
Current method :

Newton-Gauss method

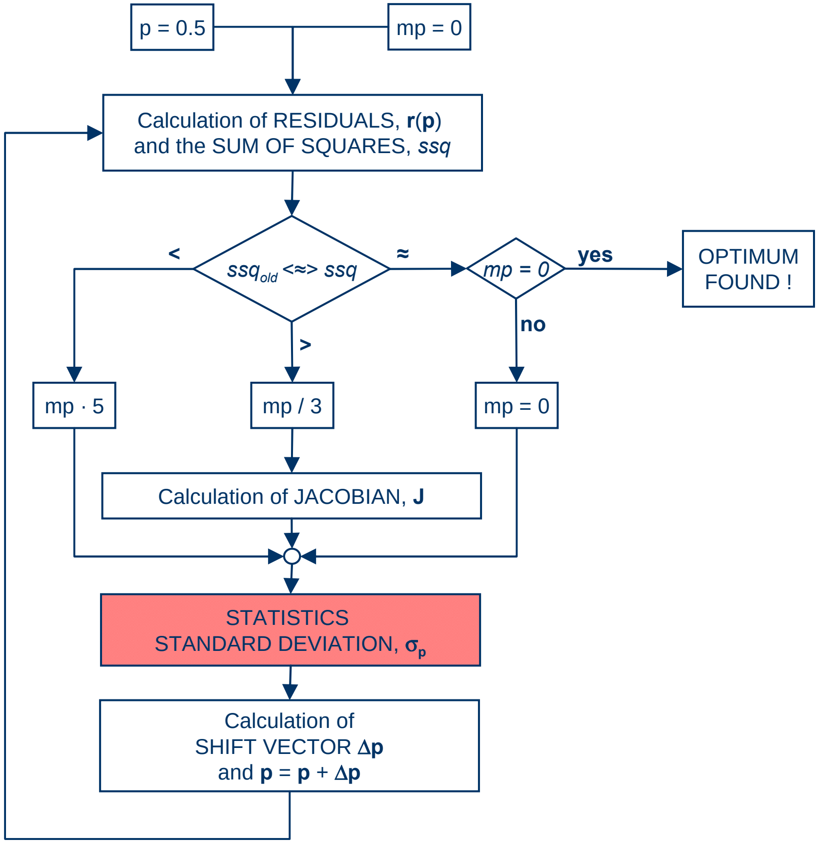


Stack: Base

Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.5	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

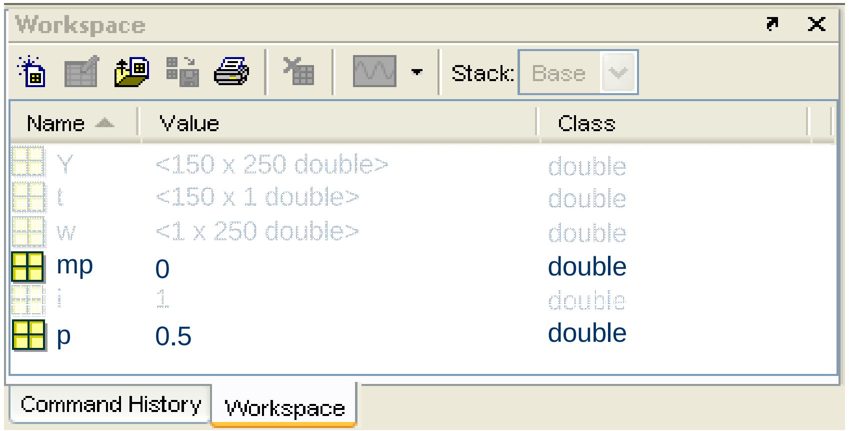
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

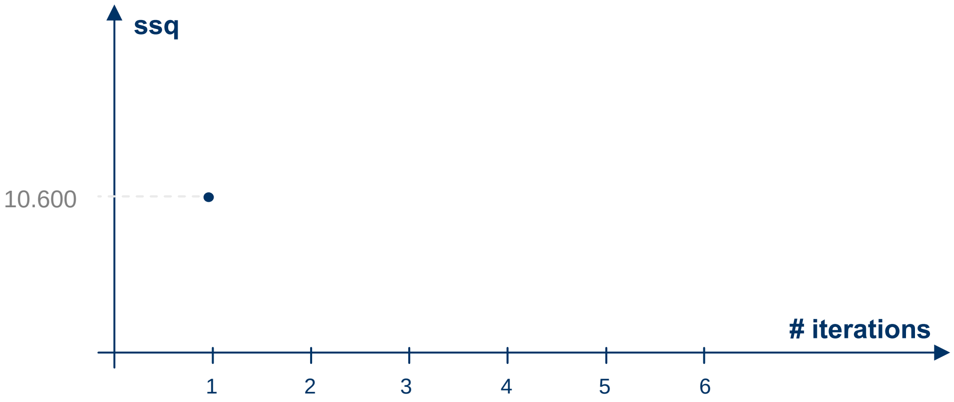
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

<u>Current method :</u>

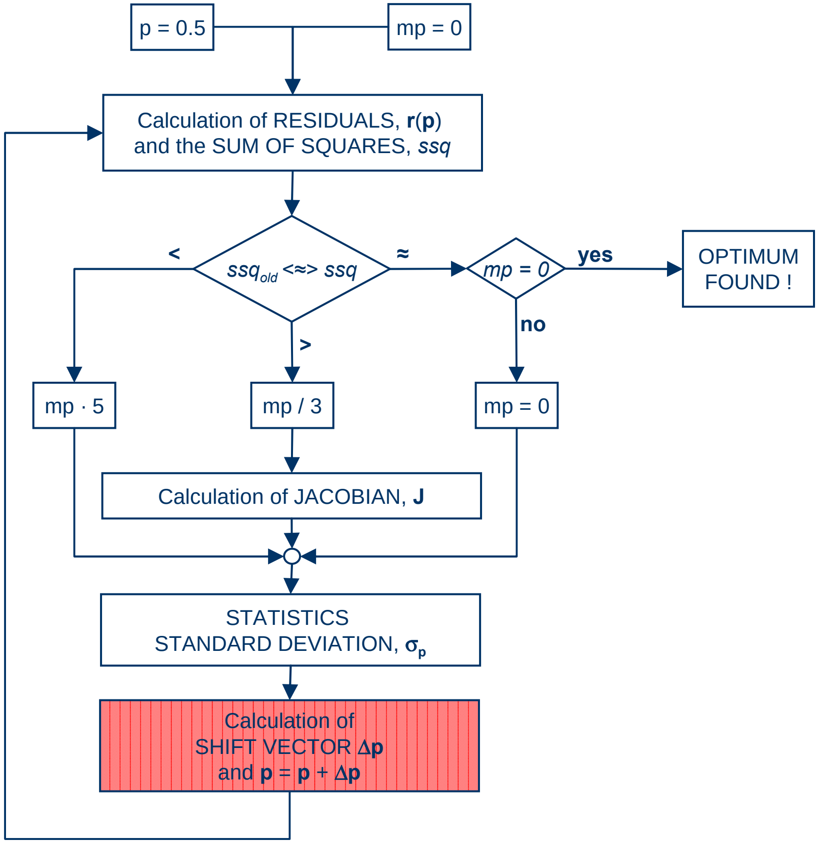
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.5	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> ssq

<

≈

>

$mp = 0$

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

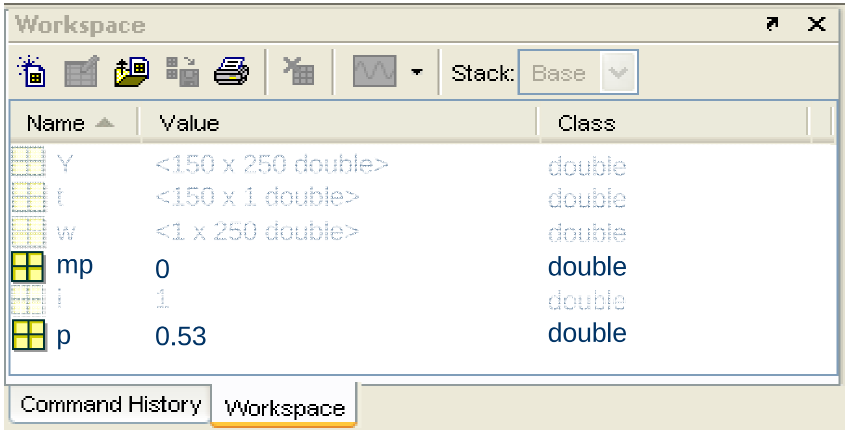
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

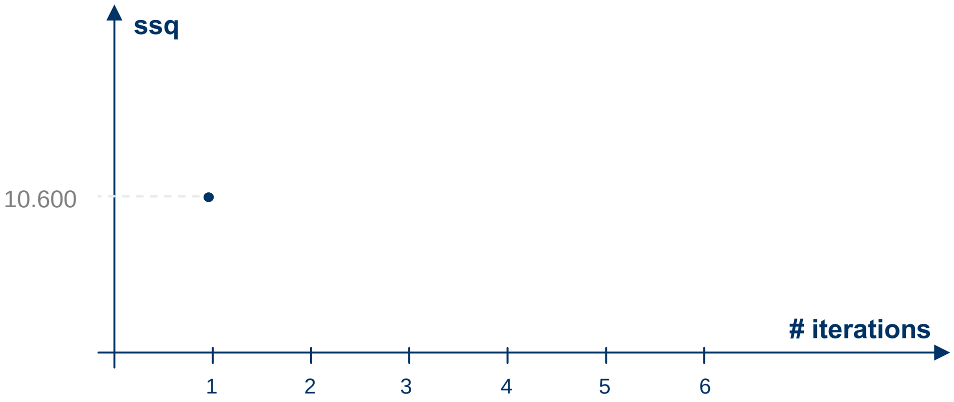
Calculation of SHIFT VECTOR **Δp** and **p** = **p** + **Δp**

Current method :

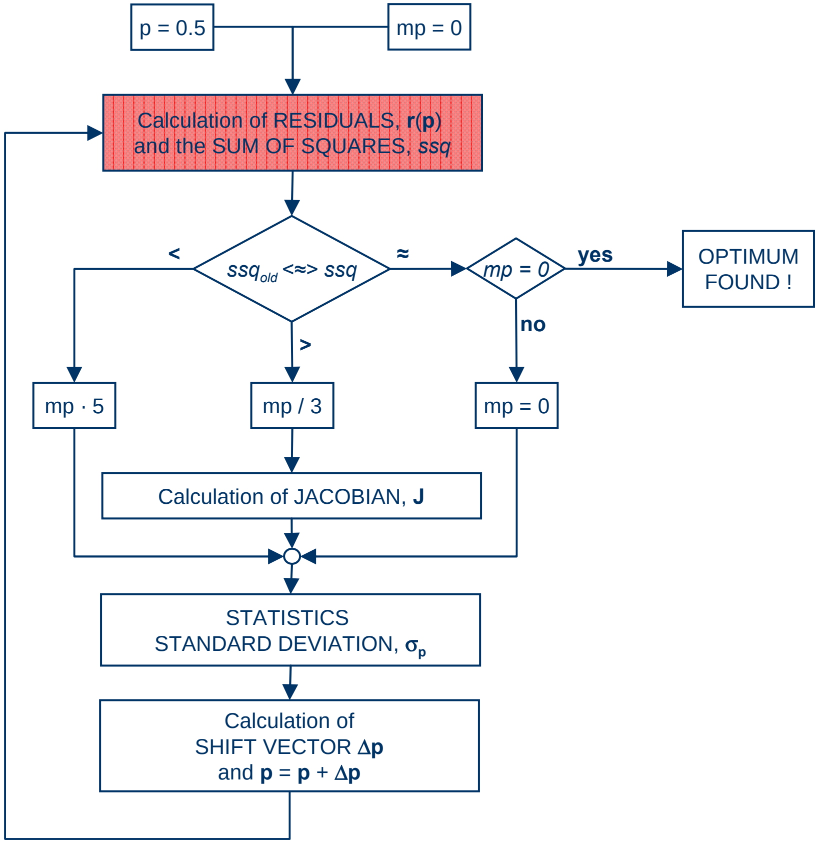
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.53	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r(p)** and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> ssq

<

≈

>

$mp = 0$

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

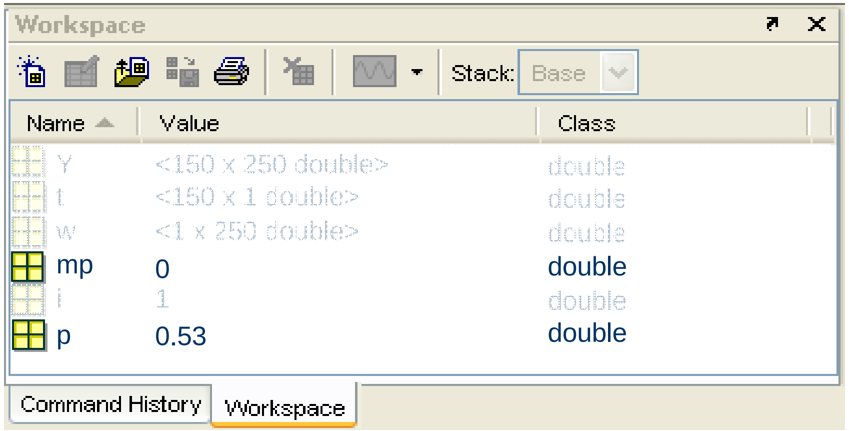
Calculation of JACOBIAN, **J**

STATISTICS
STANDARD DEVIATION, σ_p

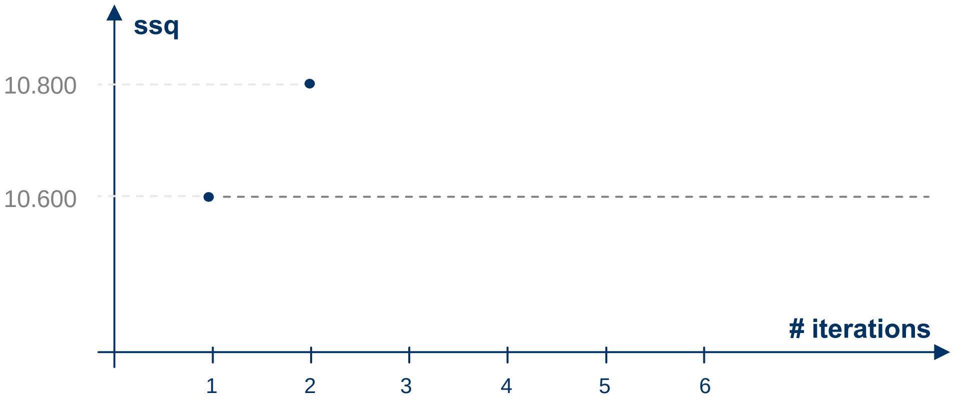
Calculation of
SHIFT VECTOR **Δp**
and **p** = **p** + **Δp**

Current method :

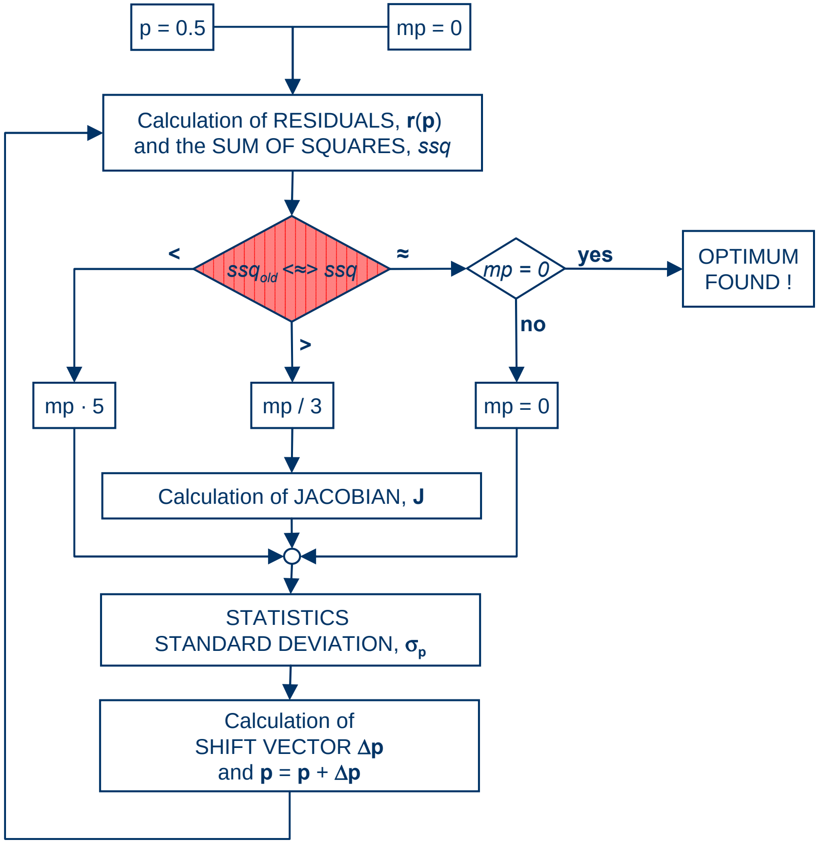
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.53	double



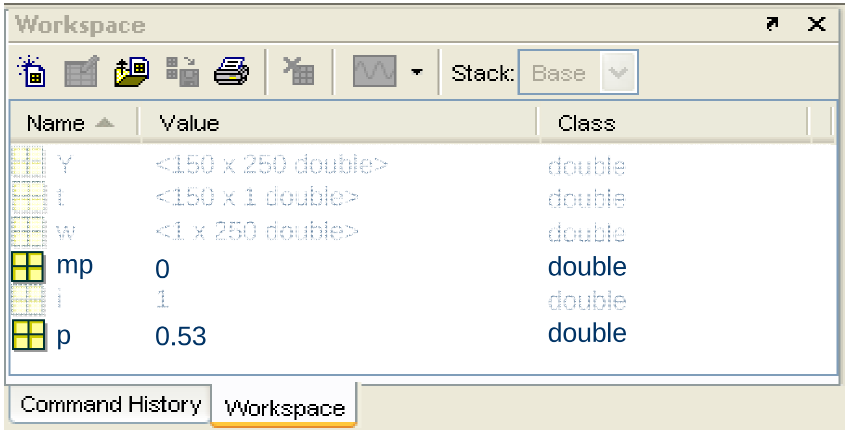
The NG/LM algorithm



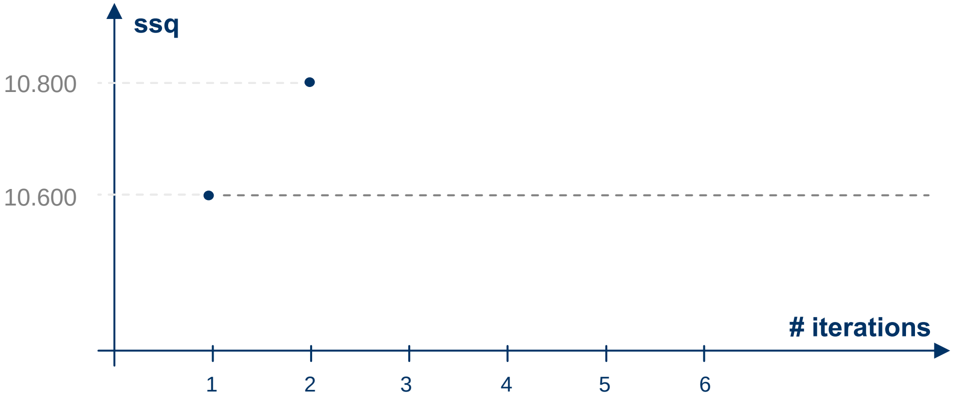
p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
<
ssq_{old} <≈> ssq
≈
mp = 0
yes
OPTIMUM FOUND !
>
no
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

<u>Current method :</u>

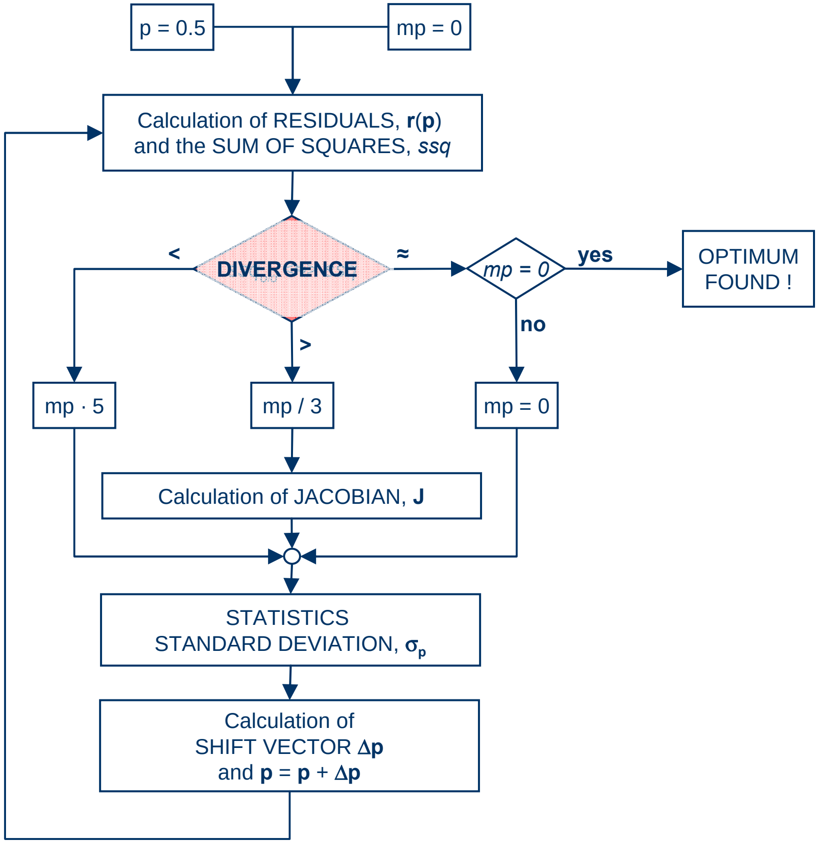
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.53	double



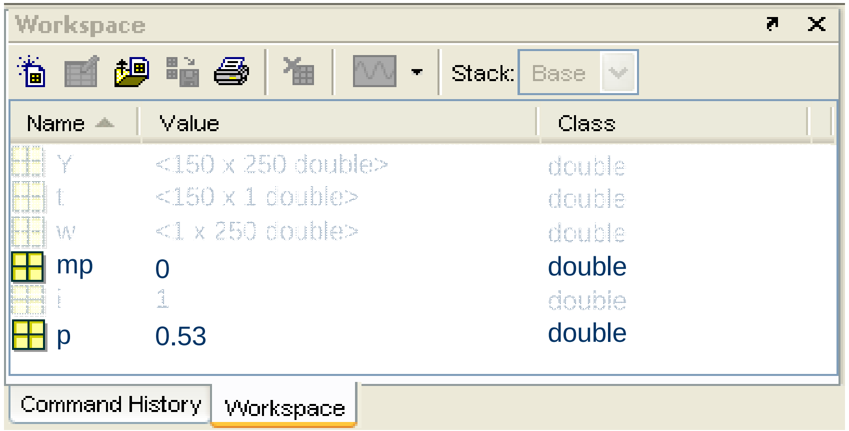
The NG/LM algorithm



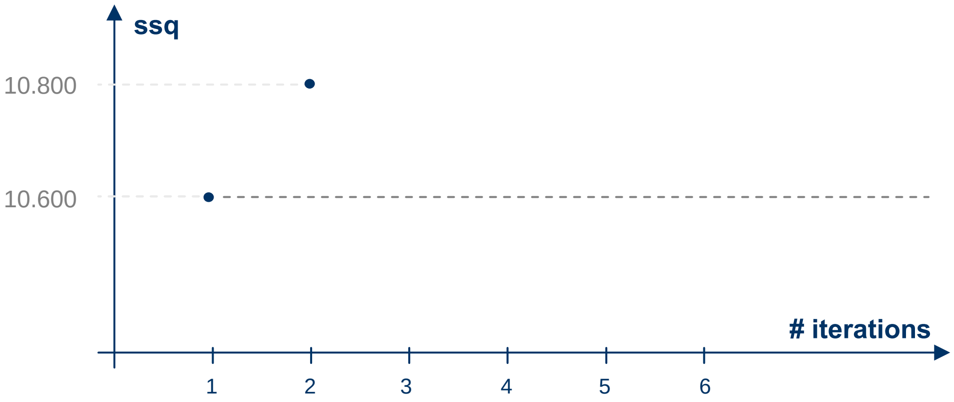
p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
DIVERGENCE
<
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

Current method :

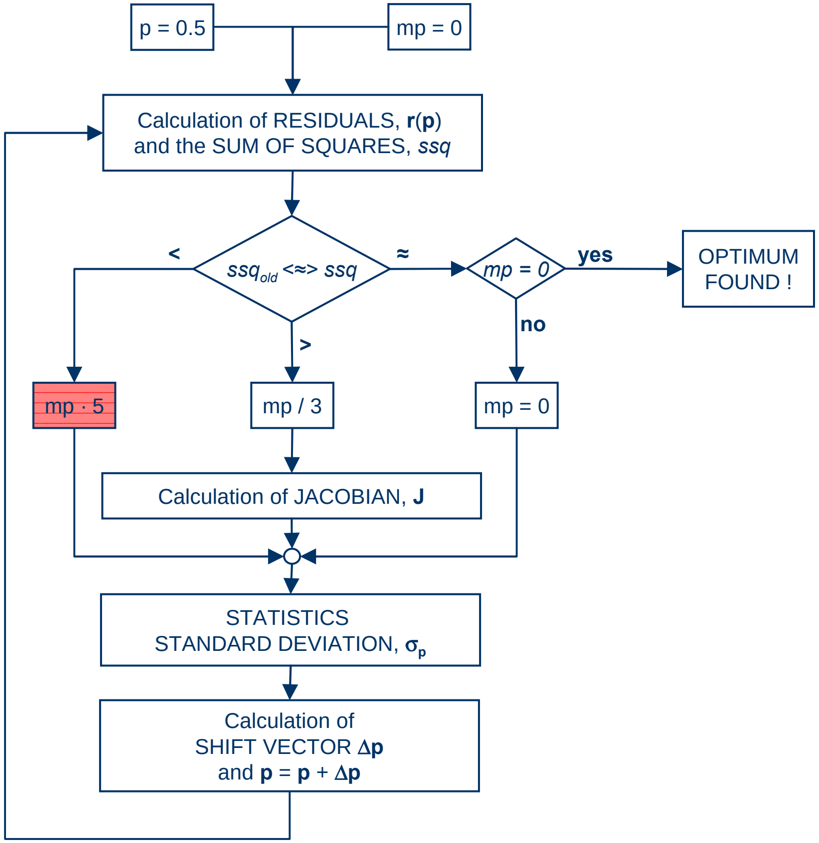
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.53	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

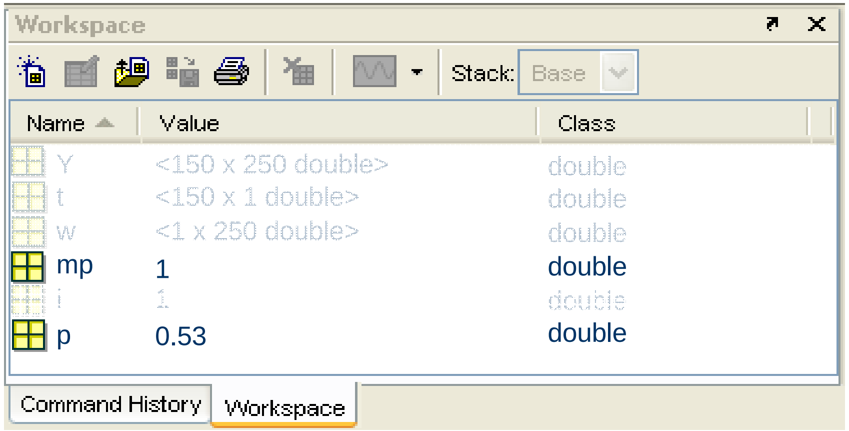
Calculation of JACOBIAN, **J**

STATISTICS
STANDARD DEVIATION, σ_p

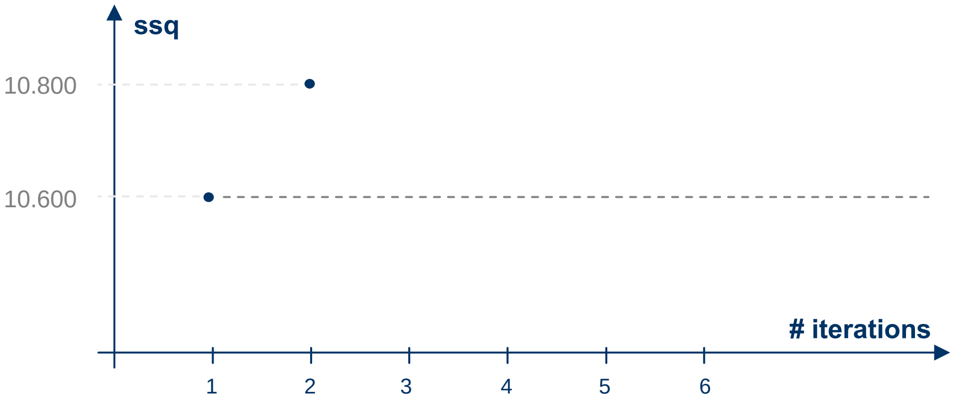
Calculation of
SHIFT VECTOR **Δp**
and **p** = **p** + **Δp**

<u>Current method :</u>

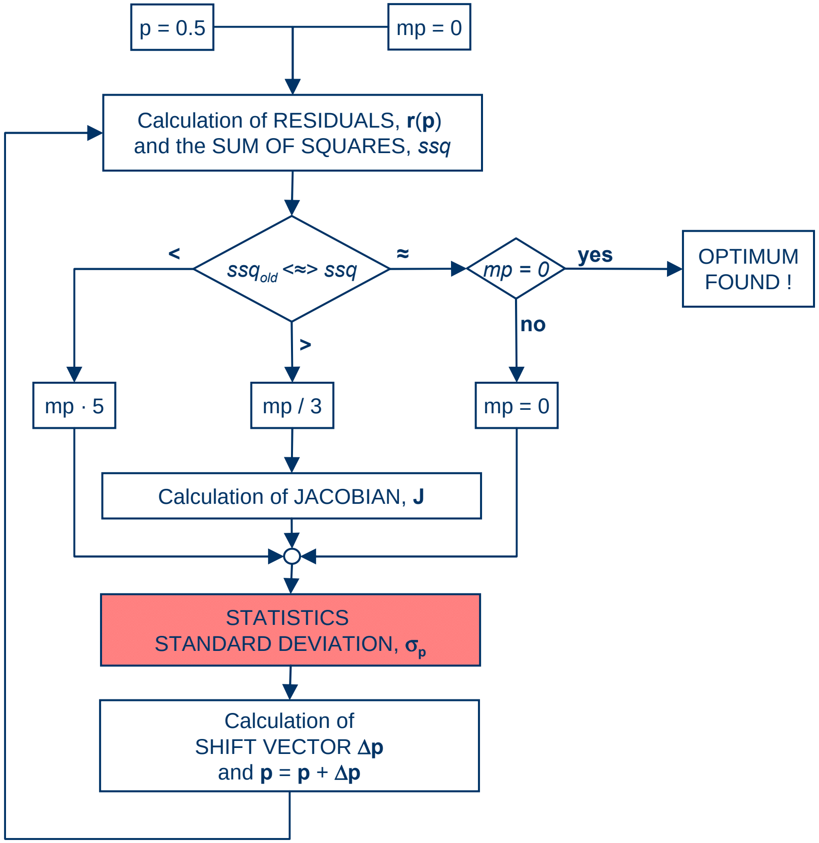
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1	double
i	1	double
p	0.53	double

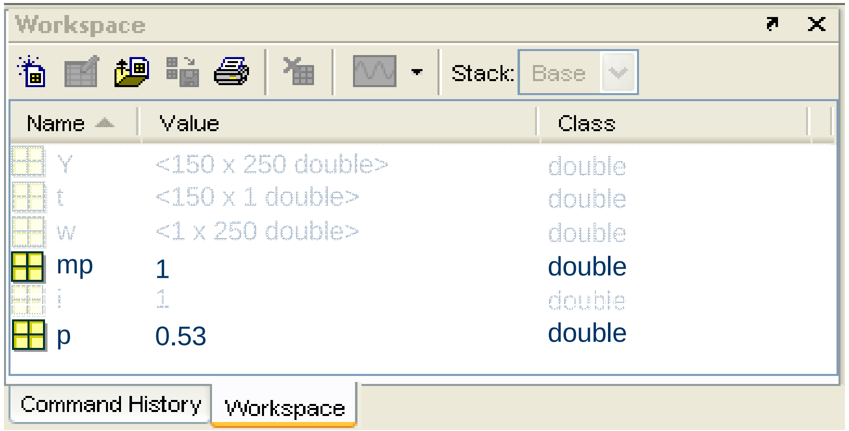


The NG/LM algorithm


p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
<
ssq_{old} <≈> *ssq*
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

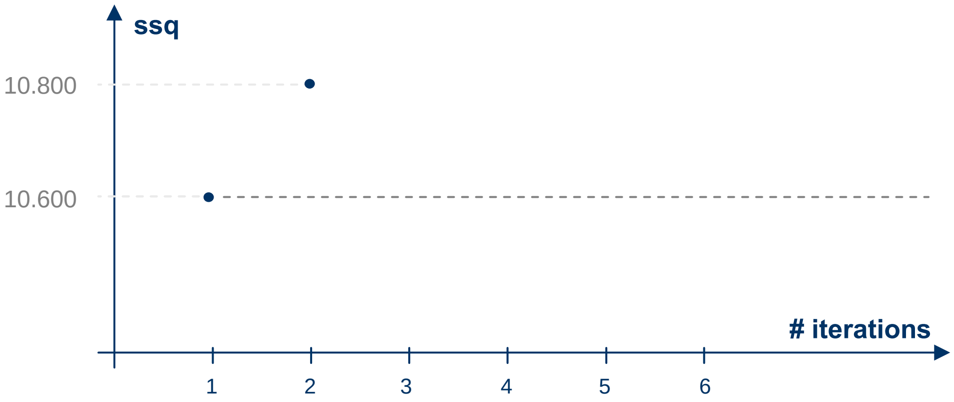
<u>Current method :</u>

NG/Levenberg-Marquardt method

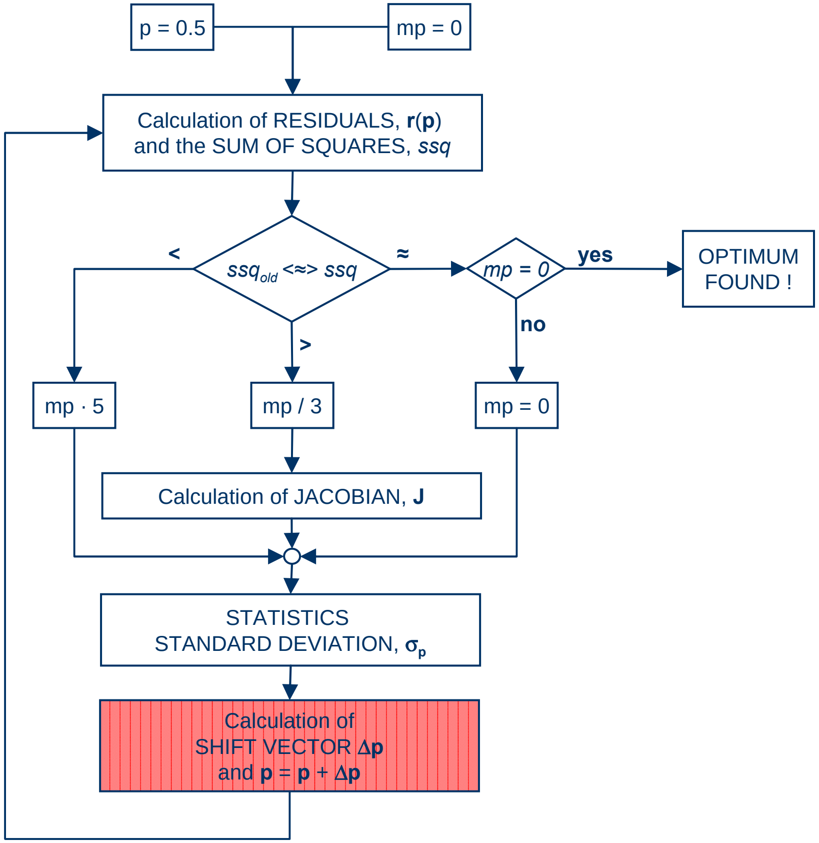


Stack: Base

Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1	double
i	1	double
p	0.53	double

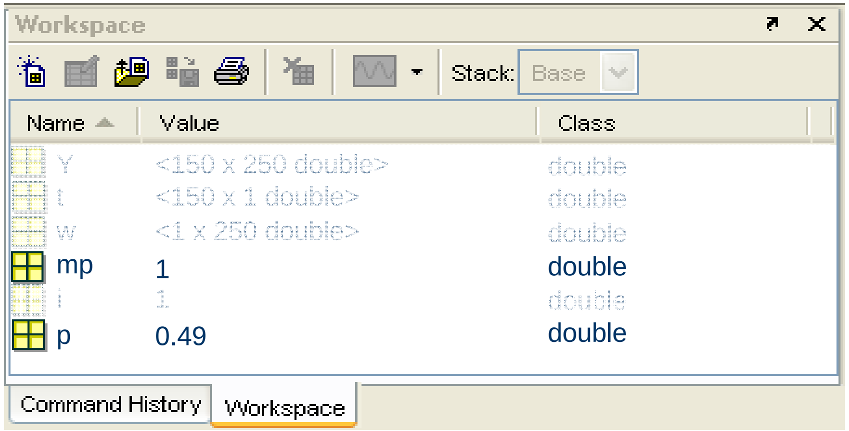


The NG/LM algorithm

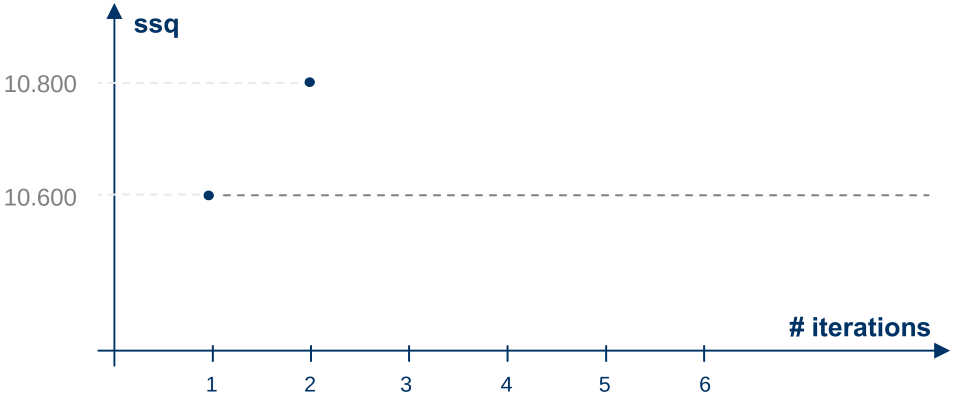

p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**)
and the SUM OF SQUARES, *ssq*
<
ssq_{old} <≈> *ssq*
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS
STANDARD DEVIATION, σ_p
Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

<u>Current method :</u>

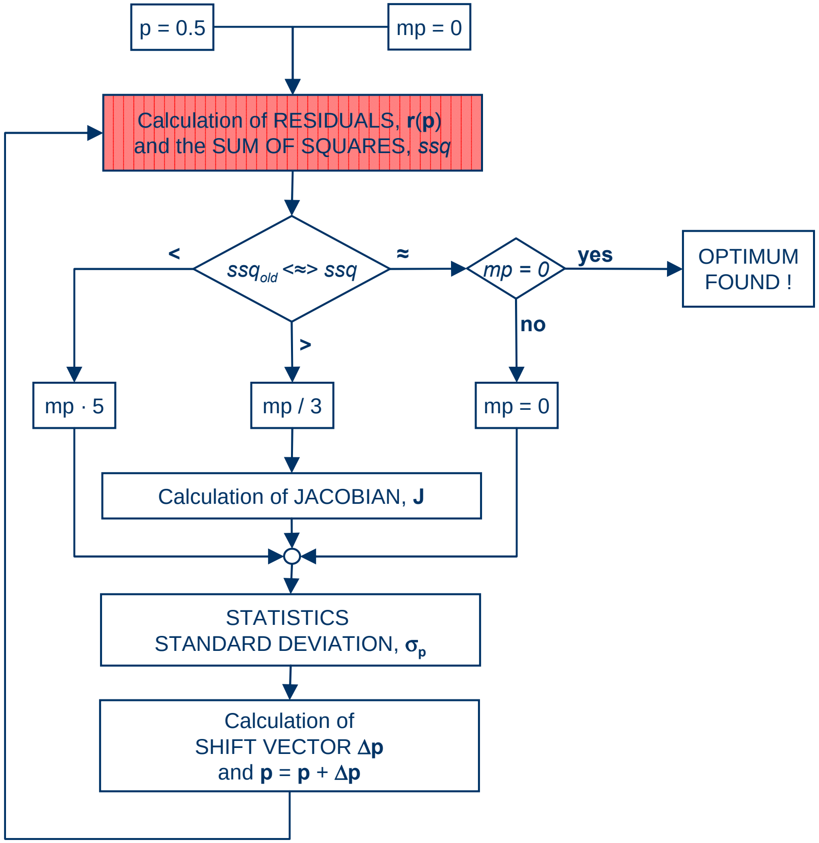
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1	double
i	1	double
p	0.49	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r(p)** and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

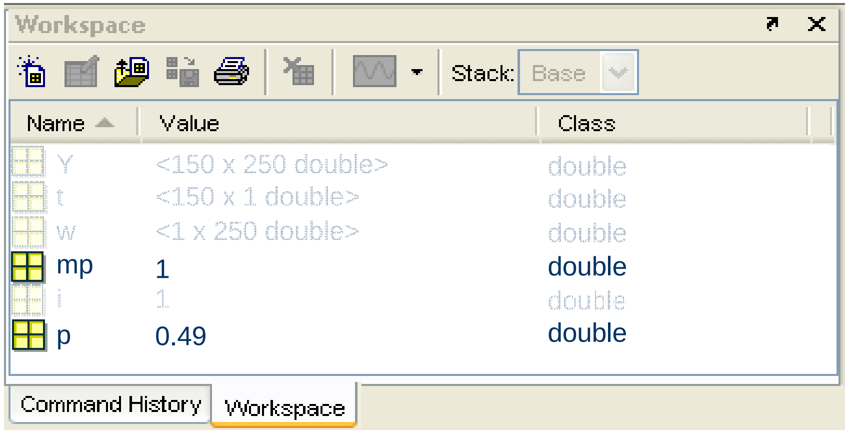
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR **Δp** and **p** = **p** + **Δp**

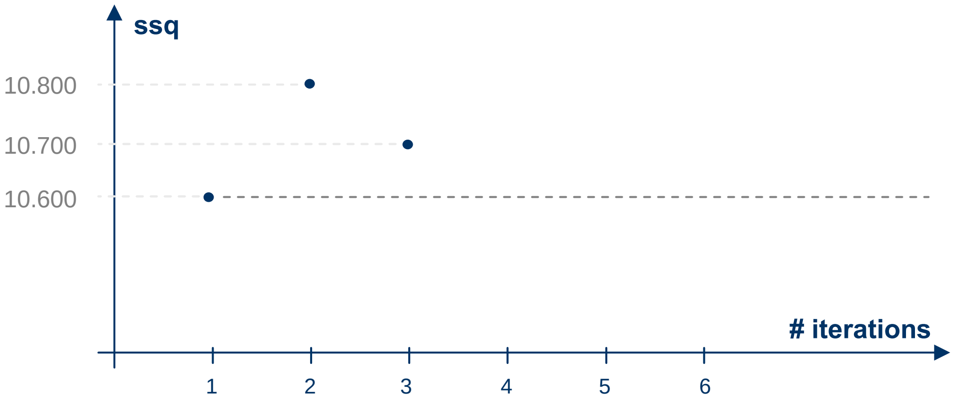
Current method :

NG/Levenberg-Marquardt method

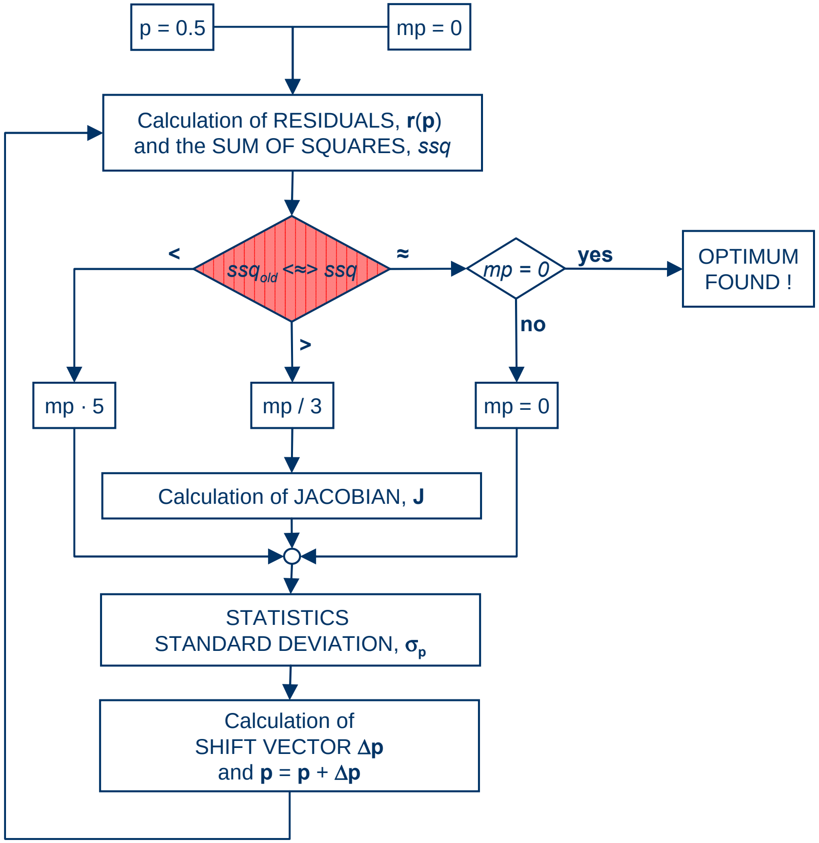


Stack: Base

Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1	double
i	1	double
p	0.49	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

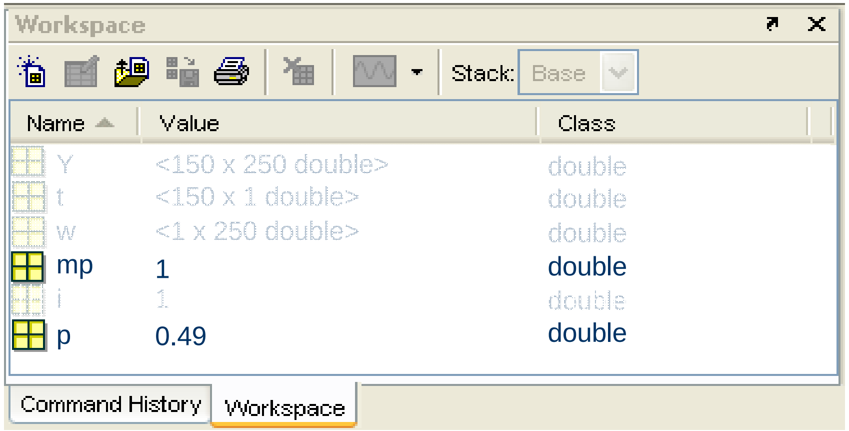
Calculation of JACOBIAN, **J**

STATISTICS
STANDARD DEVIATION, σ_p

Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

Current method :

NG/Levenberg-Marquardt method

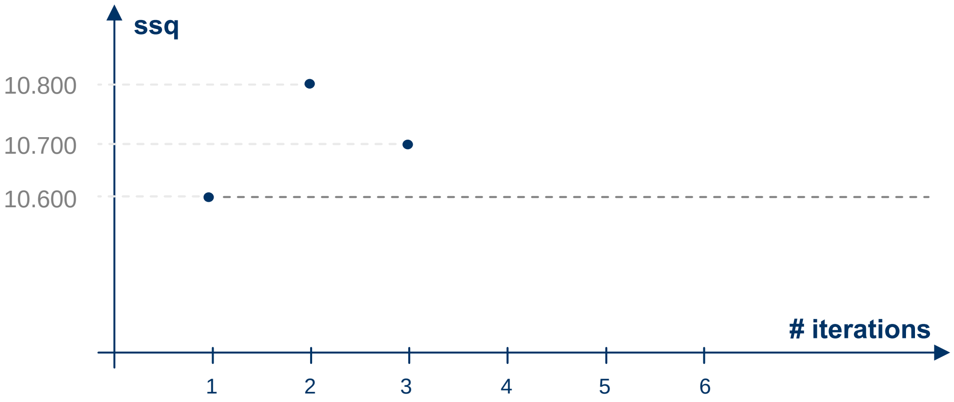


Workspace

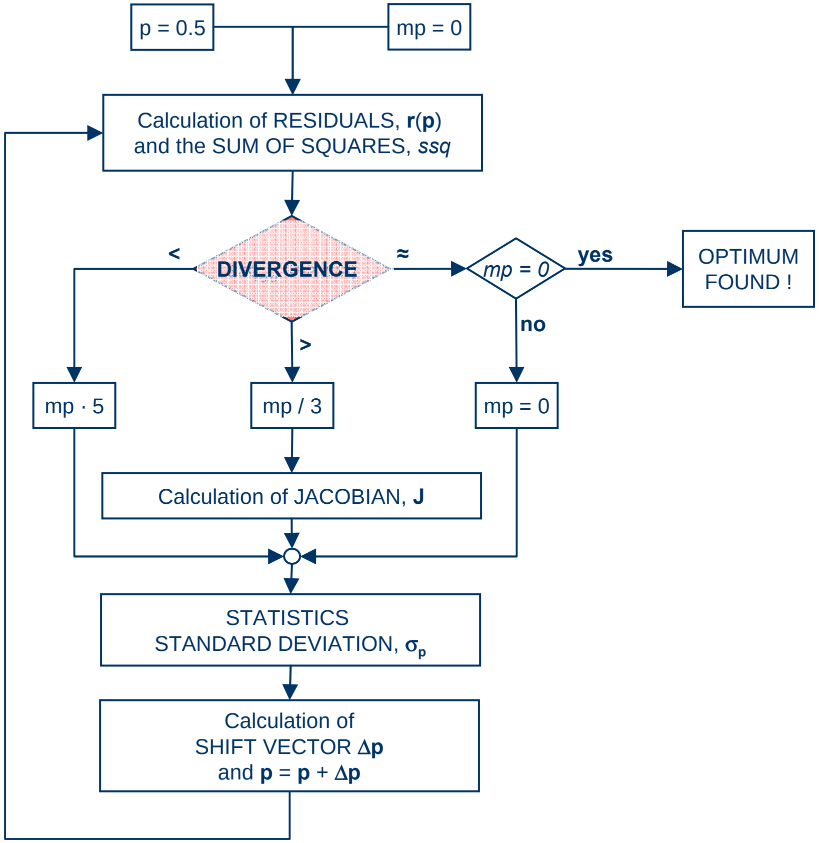
Stack: Base

Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1	double
i	1	double
p	0.49	double

Command History | Workspace

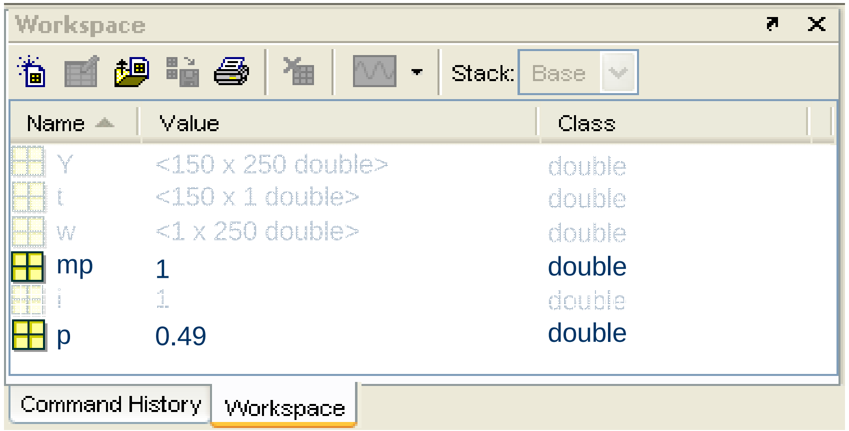


The NG/LM algorithm

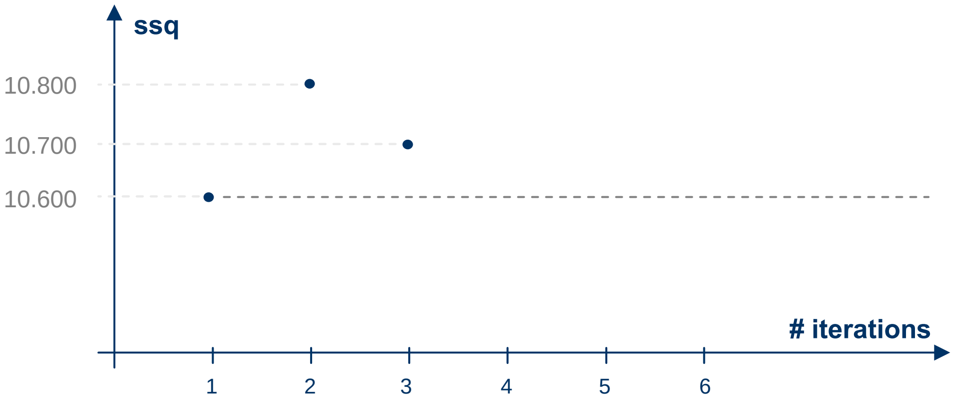

p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
DIVERGENCE
<
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δp and **p** = **p** + Δ**p**

Current method :

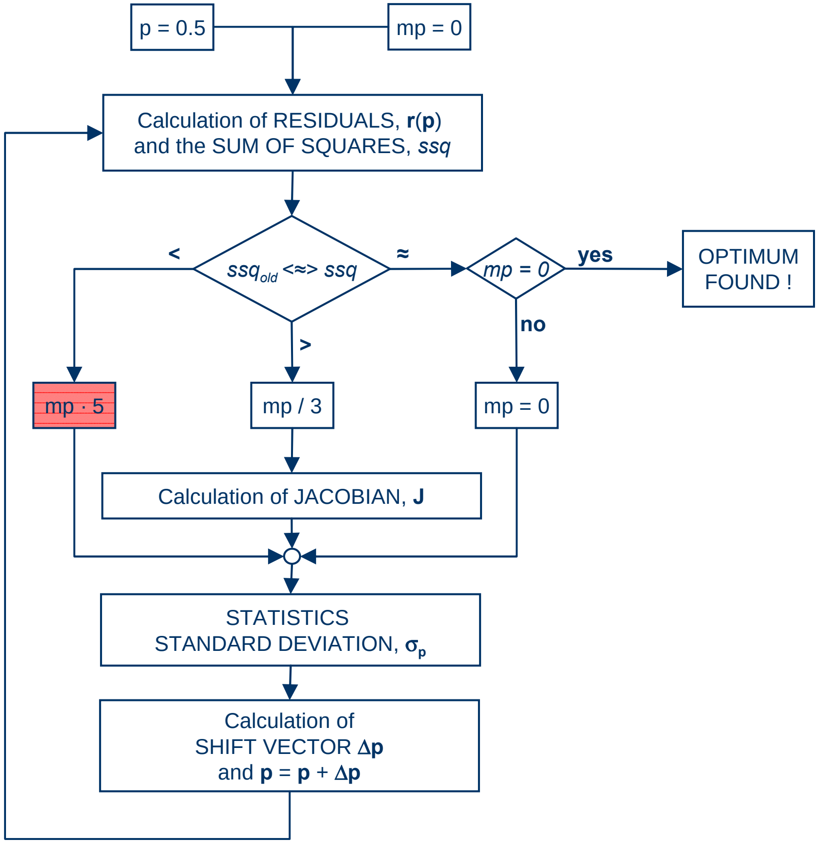
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1	double
i	1	double
p	0.49	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

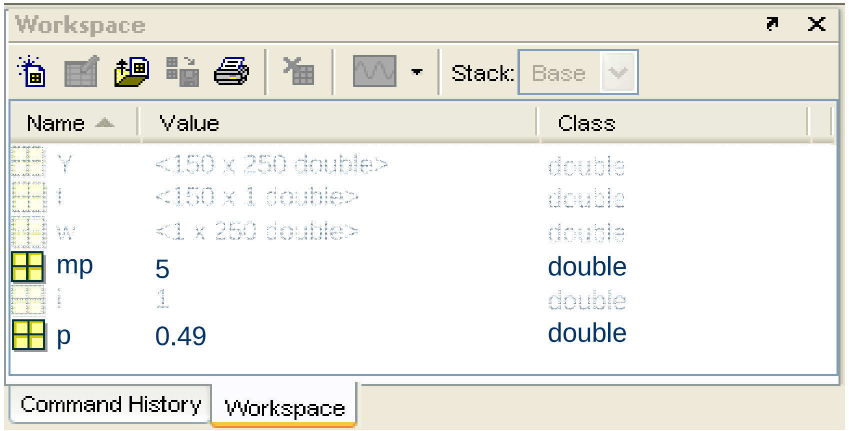
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

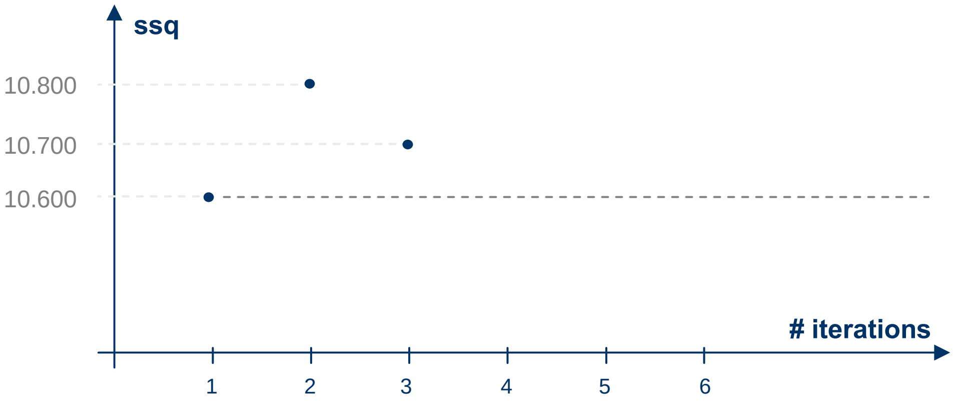
Calculation of SHIFT VECTOR **Δp** and **p** = **p** + **Δp**

Current method :

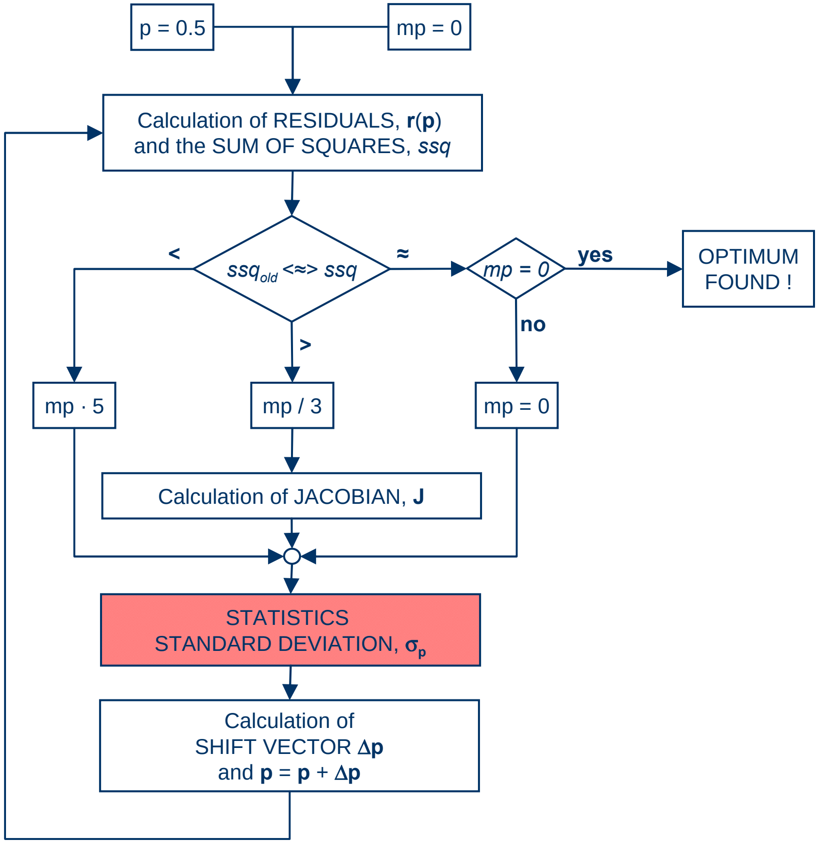
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	5	double
i	1	double
p	0.49	double



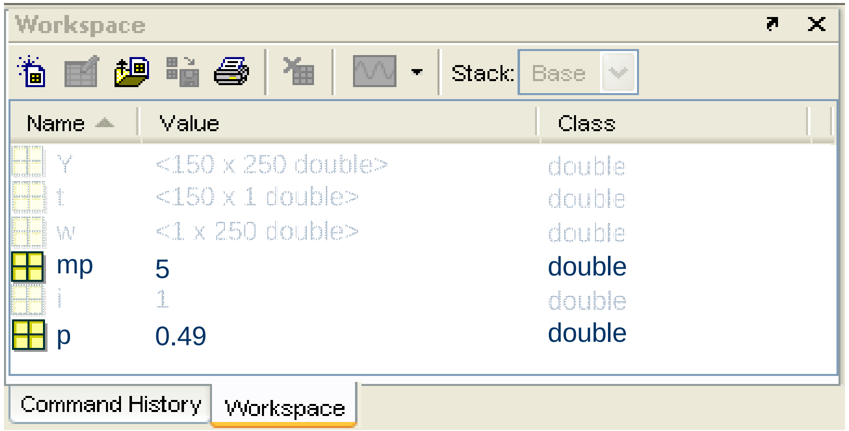
The NG/LM algorithm



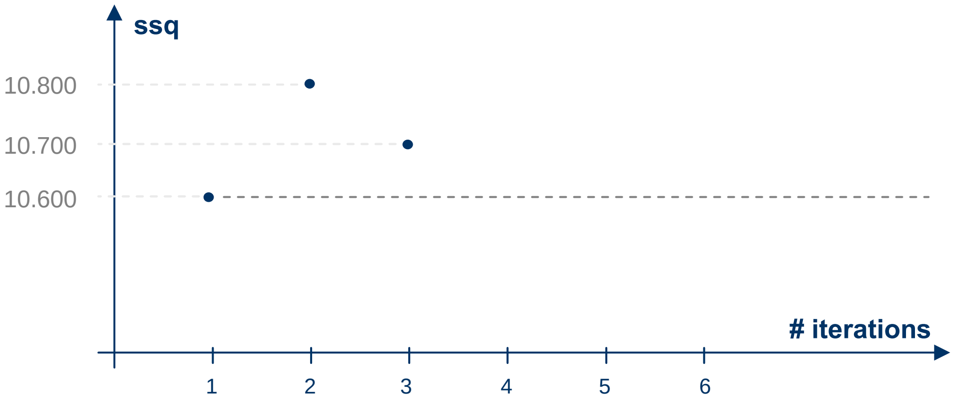
p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
ssq_{old} <≈> *ssq*
<
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

<u>Current method :</u>

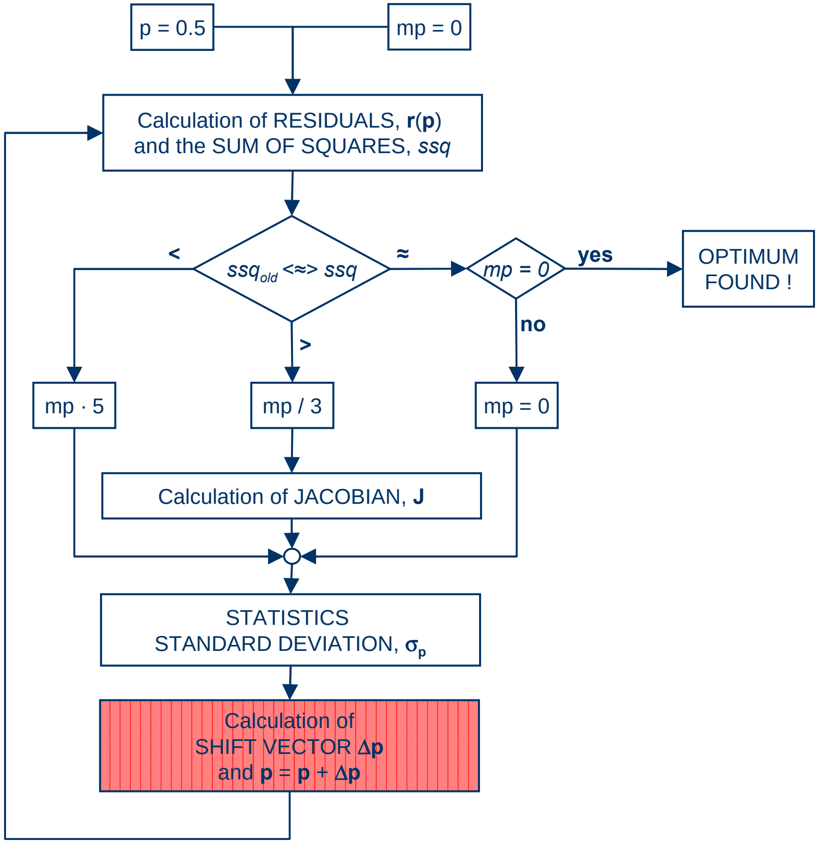
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	5	double
i	1	double
p	0.49	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

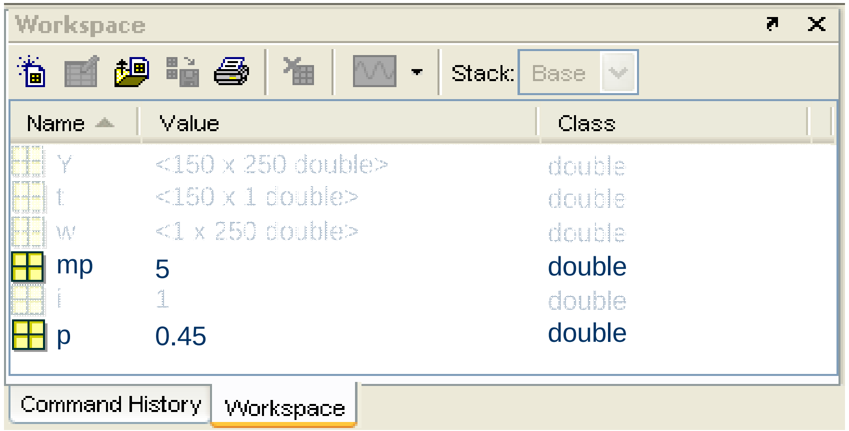
Calculation of JACOBIAN, **J**

STATISTICS
STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

Current method :

NG/Levenberg-Marquardt method

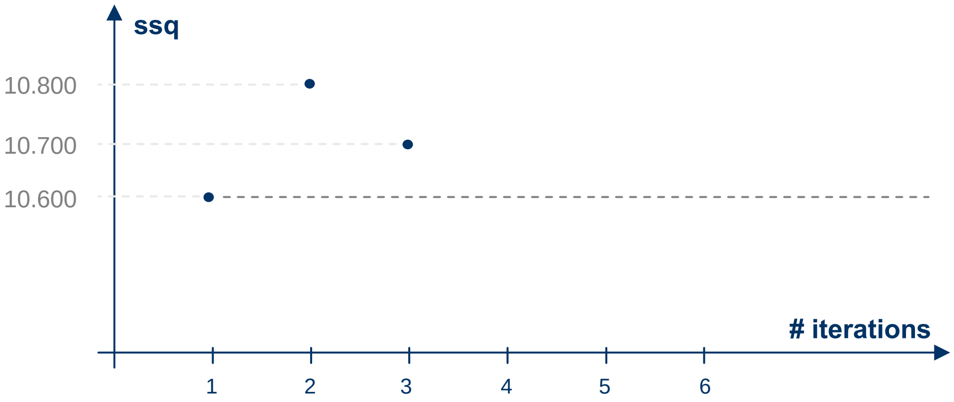


Workspace

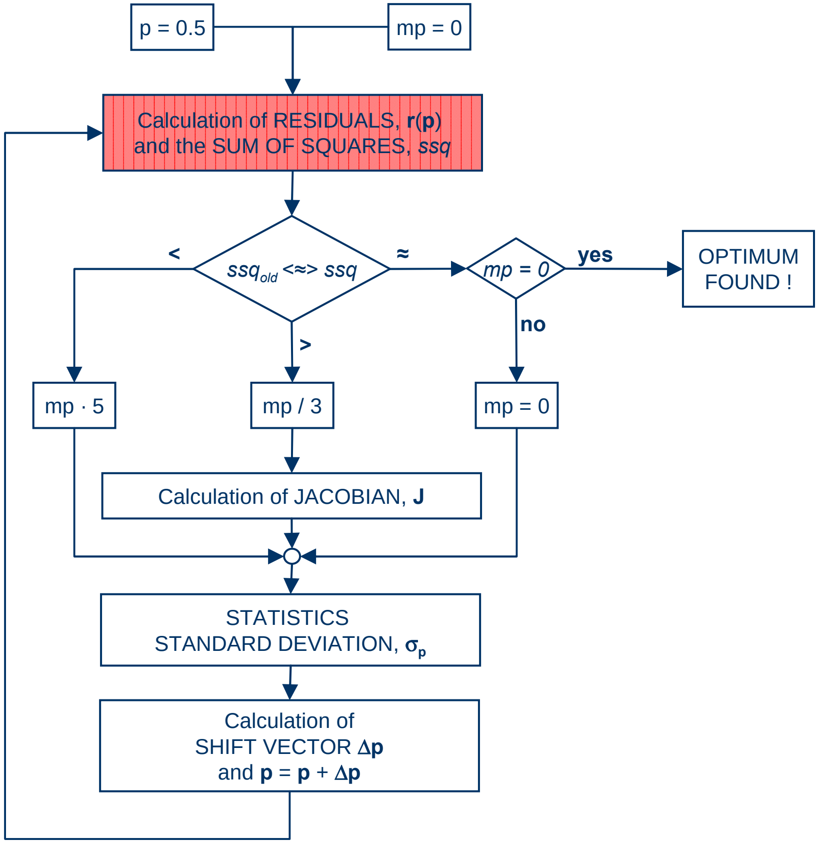
Stack: Base

Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	5	double
i	1	double
p	0.45	double

Command History | Workspace



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r(p)** and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

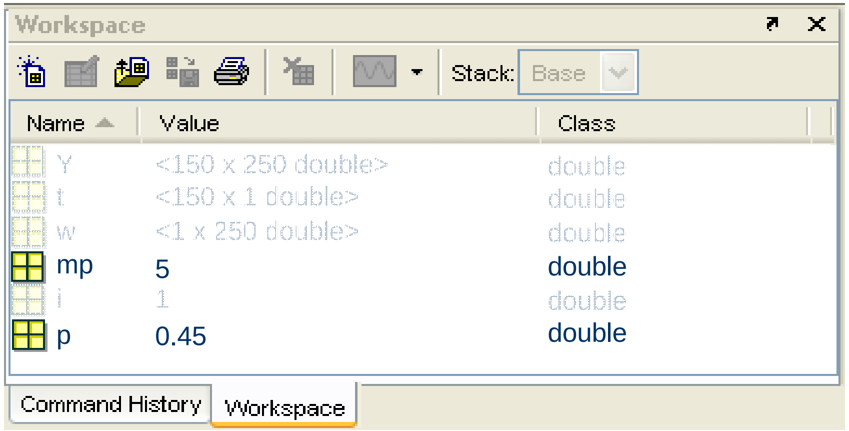
Calculation of JACOBIAN, **J**

STATISTICS
STANDARD DEVIATION, σ_p

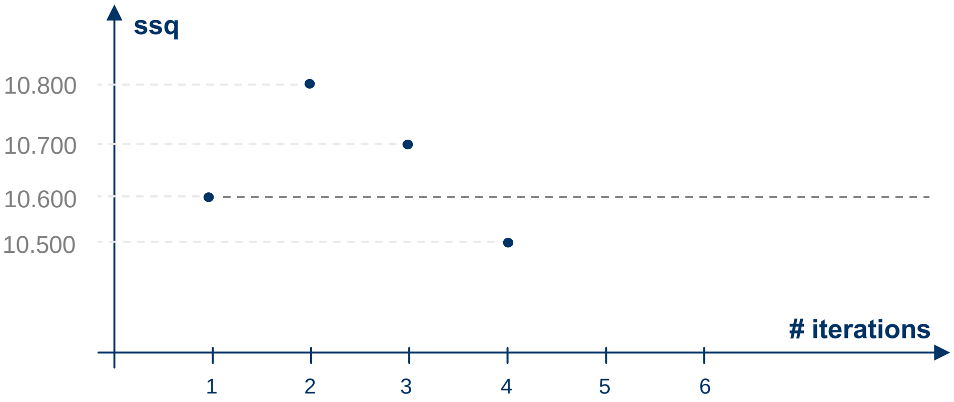
Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

Current method :

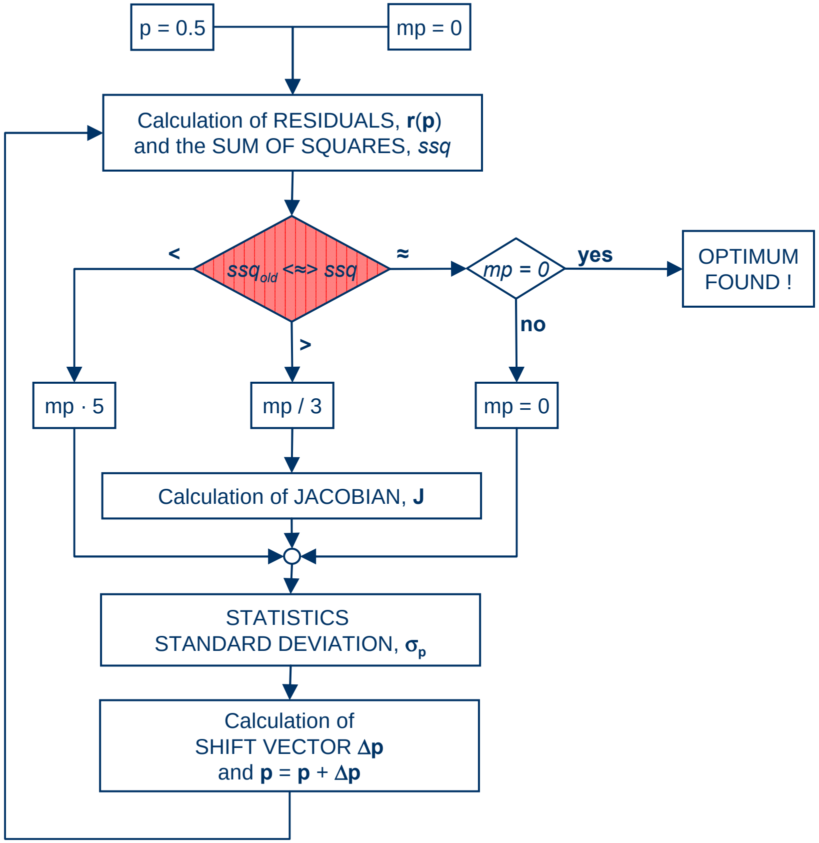
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	5	double
i	1	double
p	0.45	double



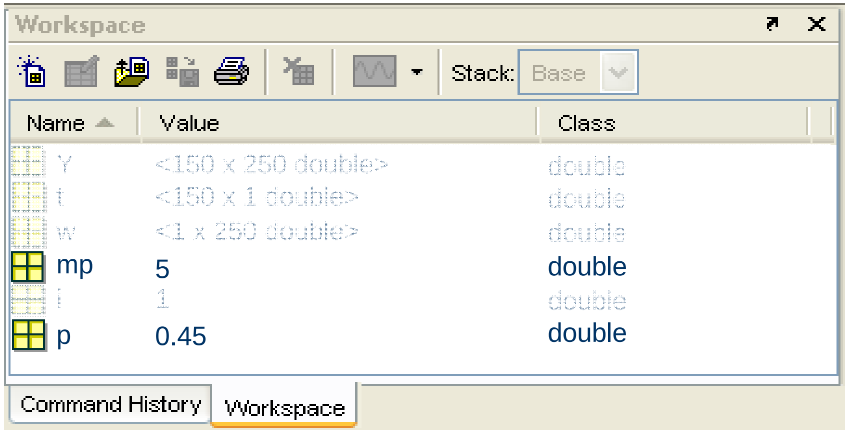
The NG/LM algorithm



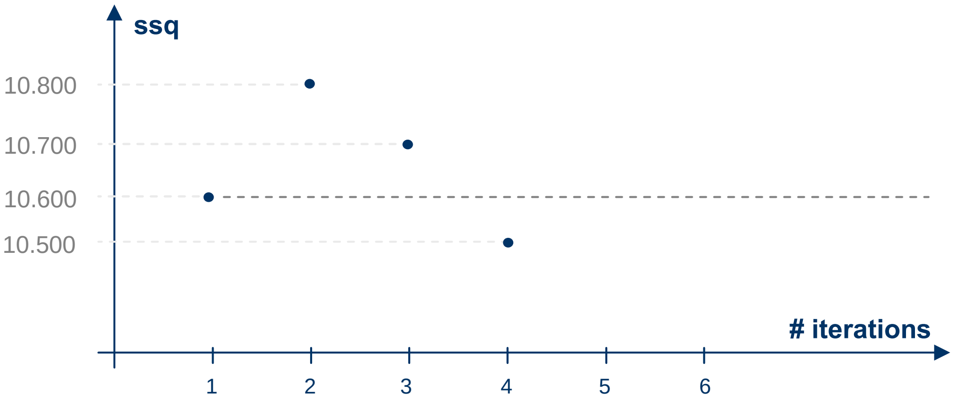
p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
ssq_{old} <≈> ssq
<
≈
>
$mp = 0$
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

<u>Current method :</u>

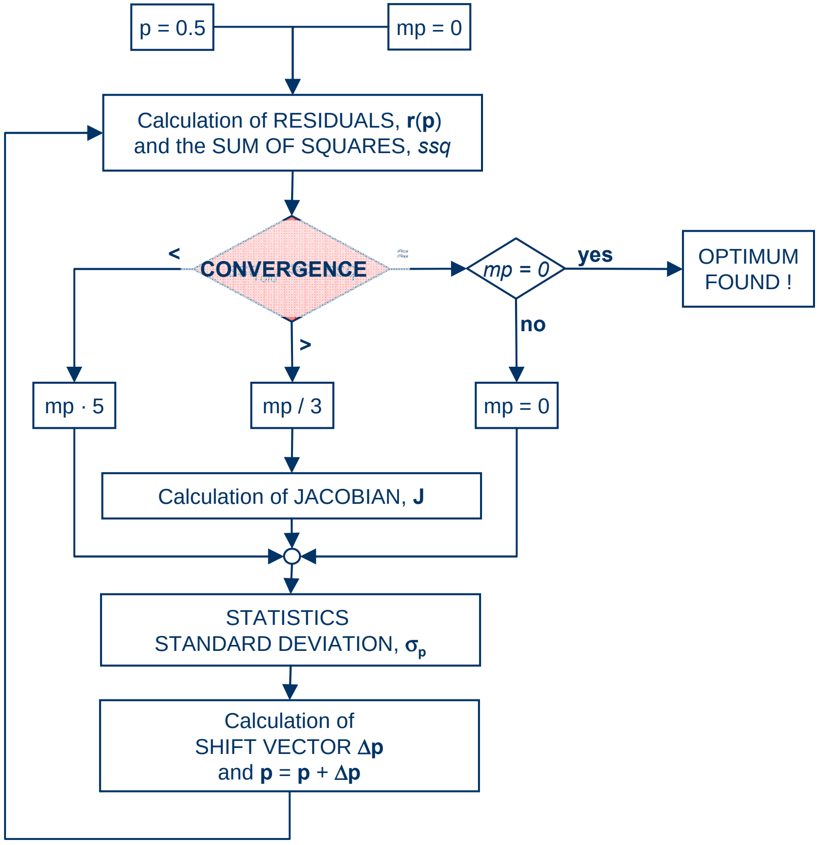
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	5	double
i	1	double
p	0.45	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

CONVERGENCE

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

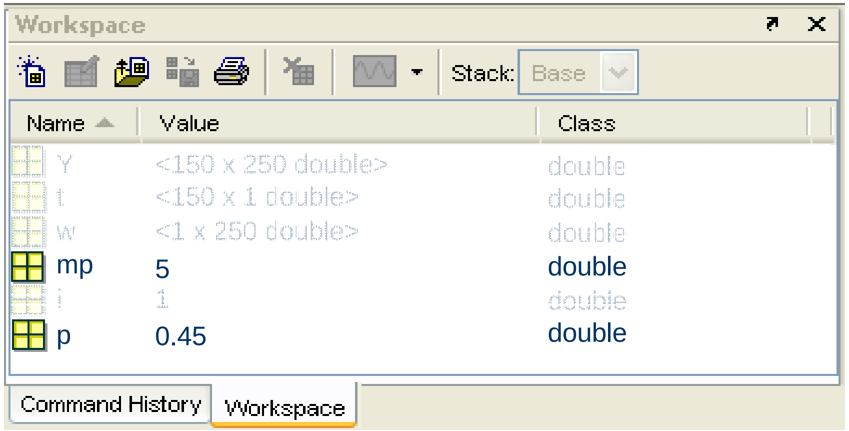
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

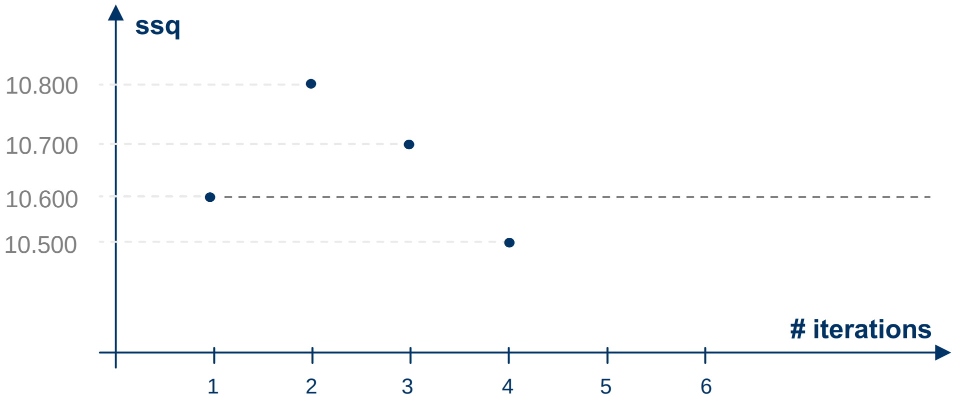
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

Current method :

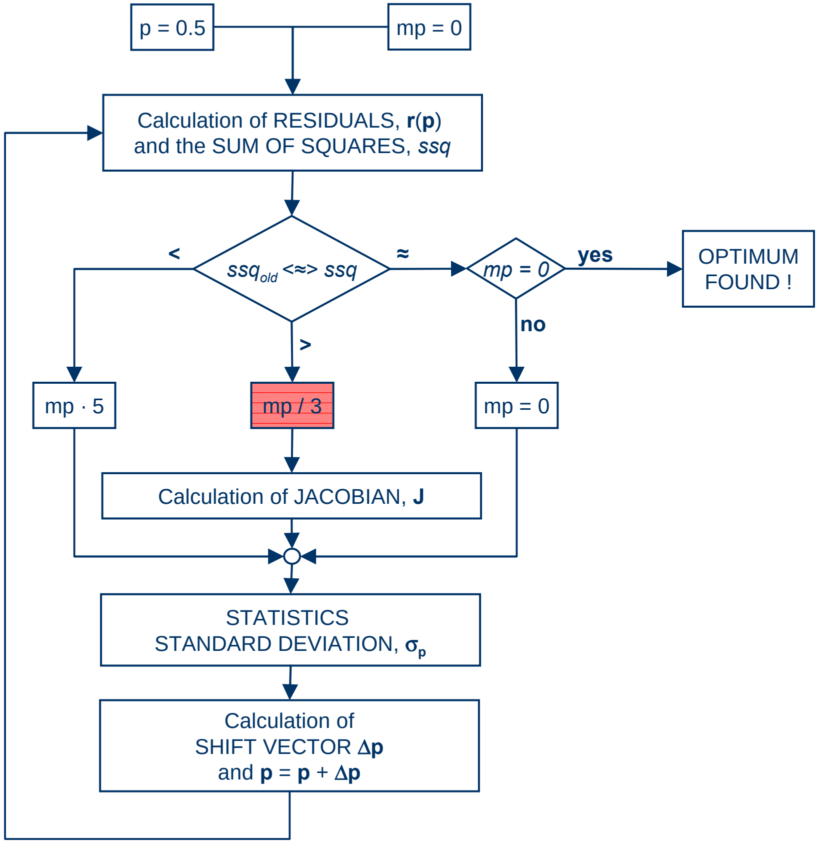
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	5	double
i	1	double
p	0.45	double

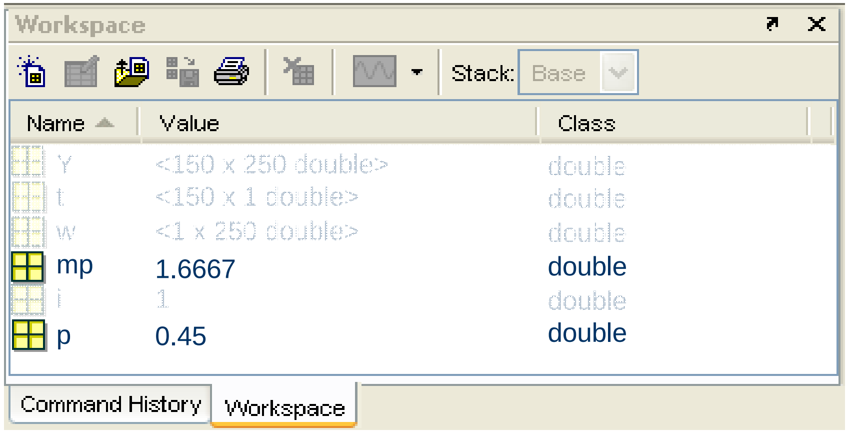


The NG/LM algorithm

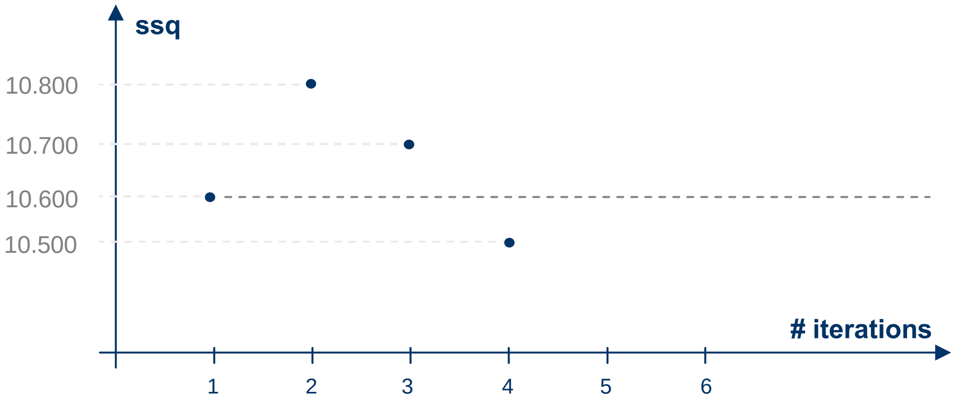

p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**)
and the SUM OF SQUARES, *ssq*
ssq_{old} <≈> *ssq*
<
≈
>
mp = 0
yes
no
OPTIMUM
FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS
STANDARD DEVIATION, σ_p
Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

Current method :

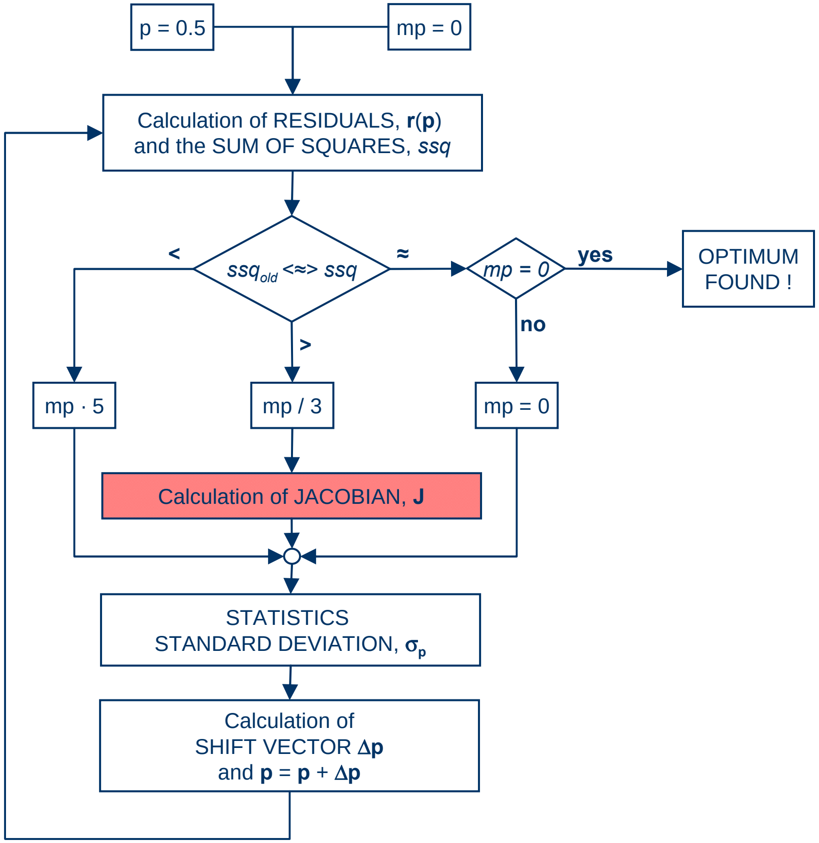
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1.6667	double
i	1	double
p	0.45	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

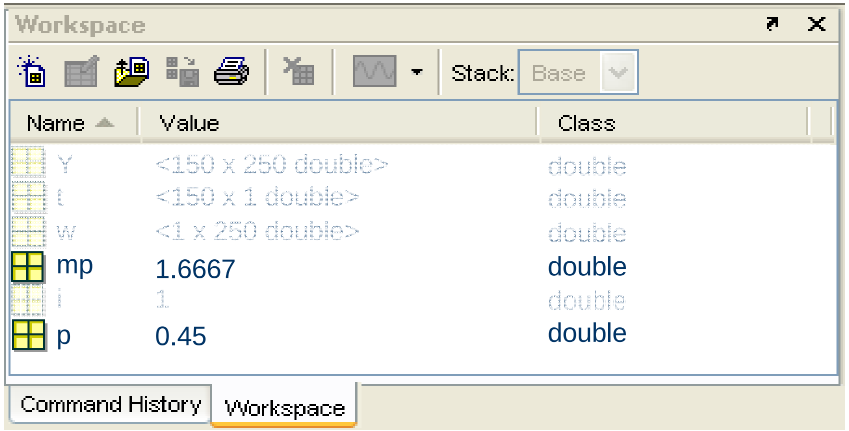
Calculation of JACOBIAN, **J**

STATISTICS
STANDARD DEVIATION, σ_p

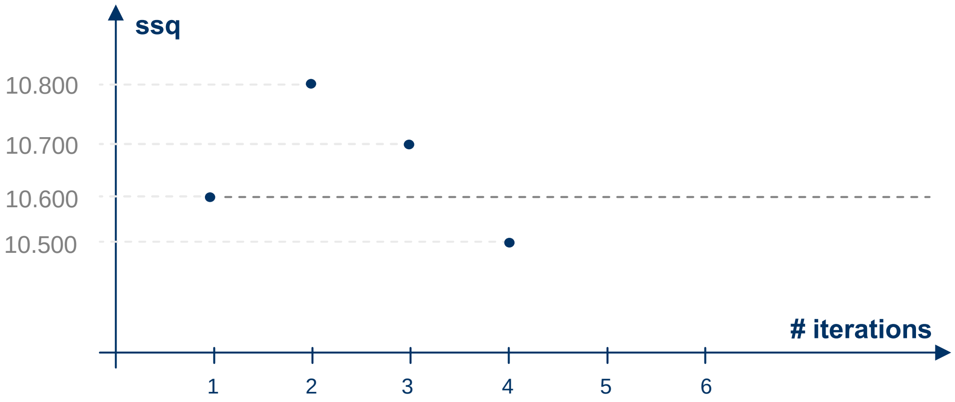
Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

<u>Current method :</u>

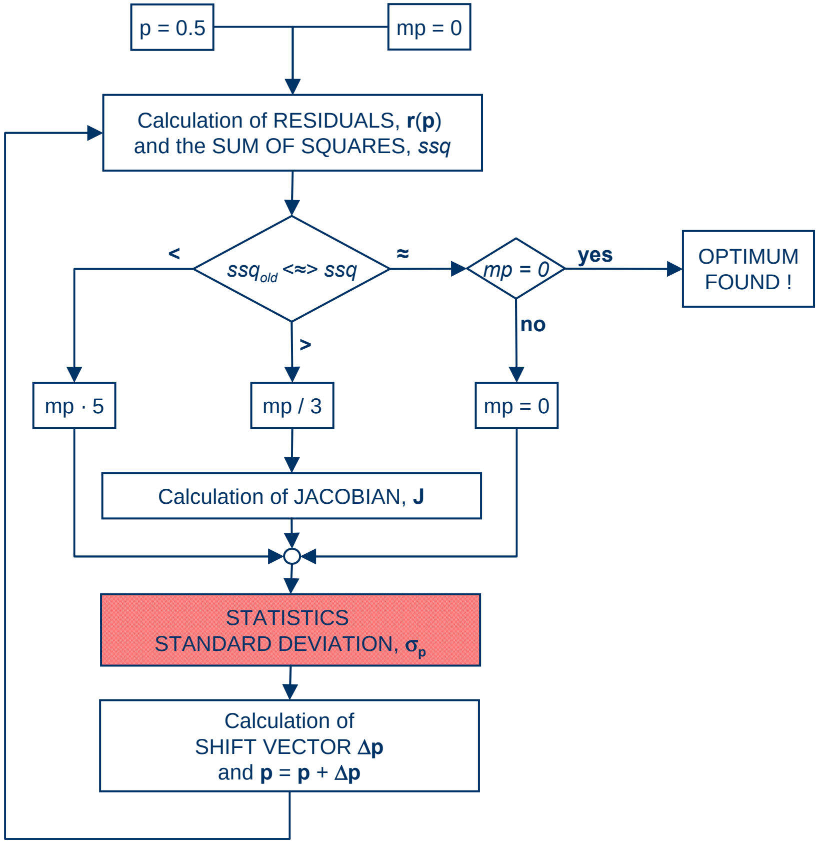
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1.6667	double
i	1	double
p	0.45	double

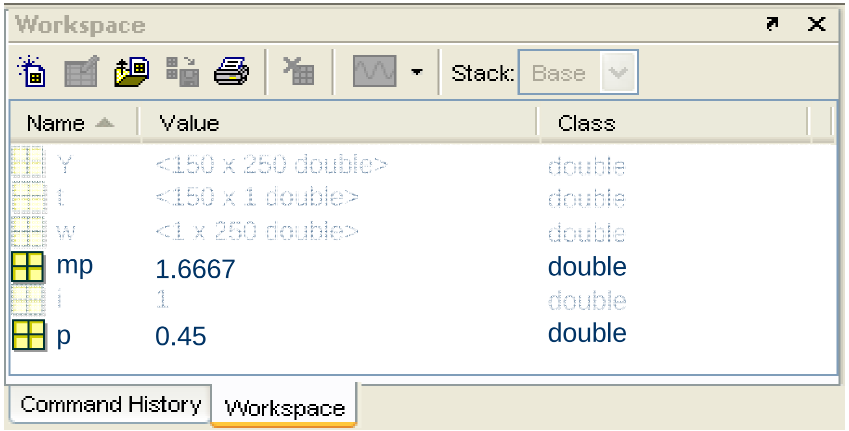


The NG/LM algorithm

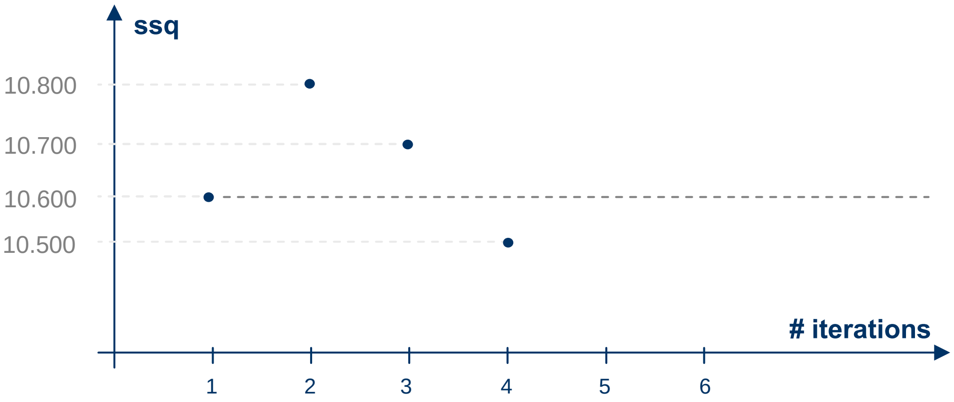

p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
<
ssq_{old} <≈> *ssq*
≈
mp = 0
yes
OPTIMUM FOUND !
>
no
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

Current method :

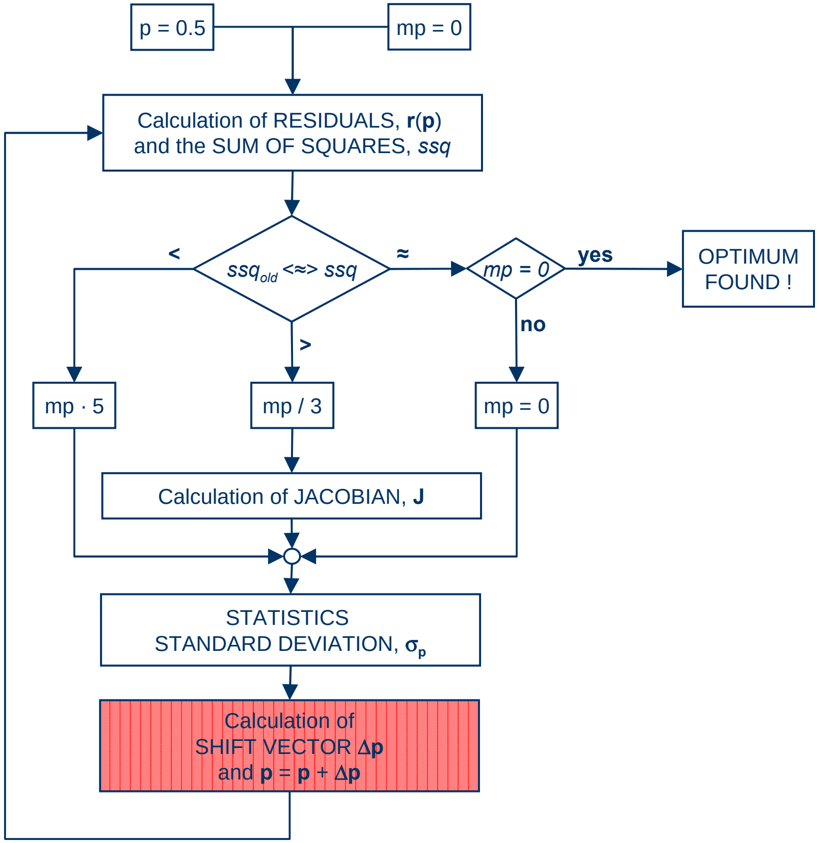
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1.6667	double
i	1	double
p	0.45	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

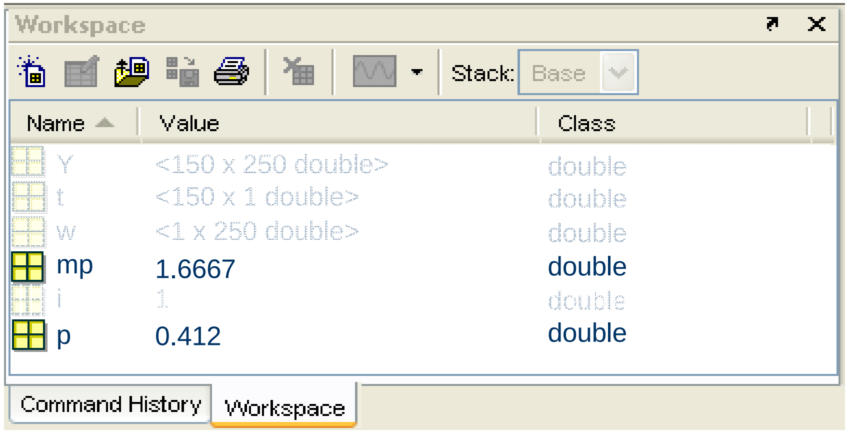
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

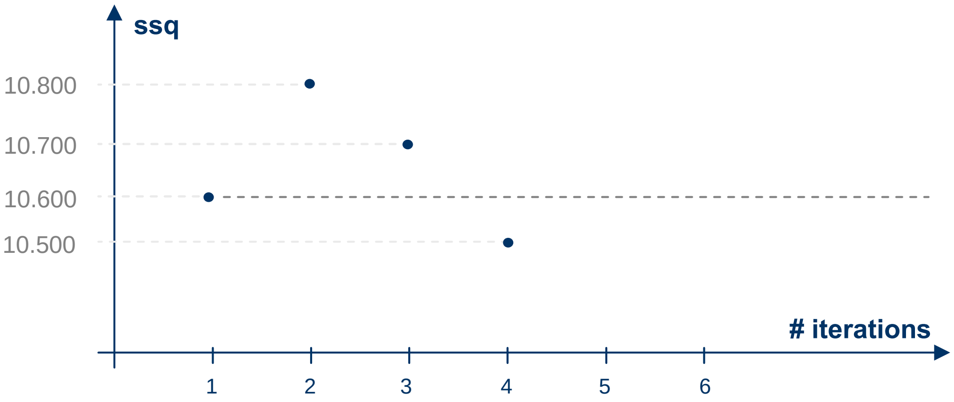
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

Current method :

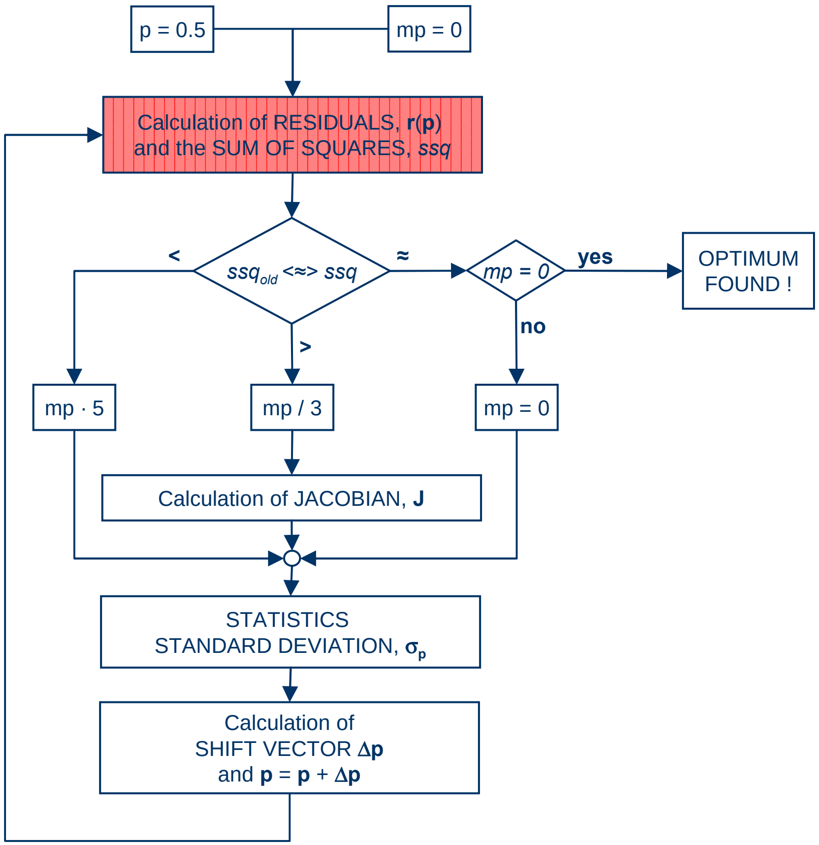
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1.6667	double
i	1	double
p	0.412	double



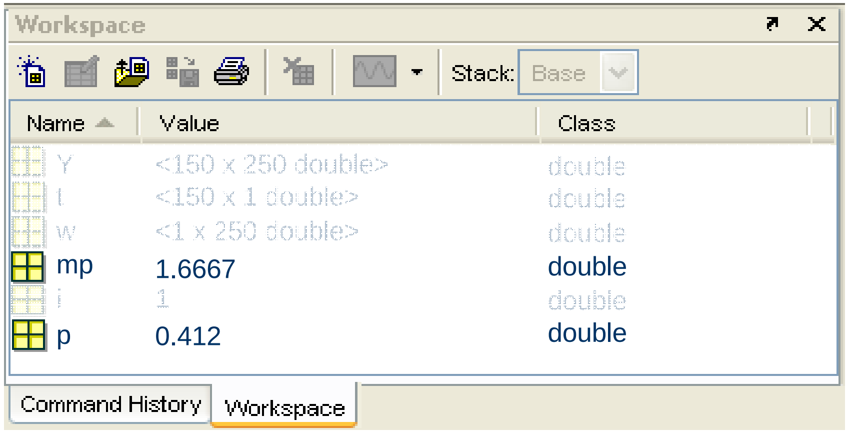
The NG/LM algorithm



$$\frac{10.500-10.499}{10.500}=9.523\cdot10^{-5}<10^{-4}$$

Current method :

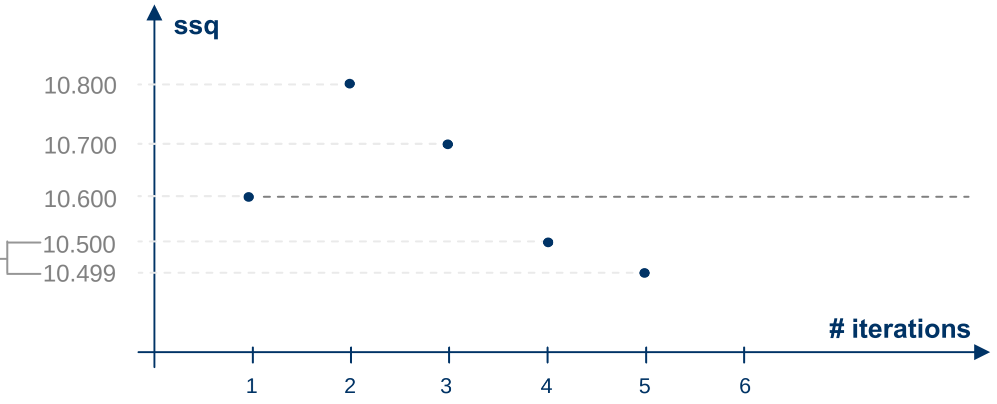
NG/Levenberg-Marquardt method


Workspace

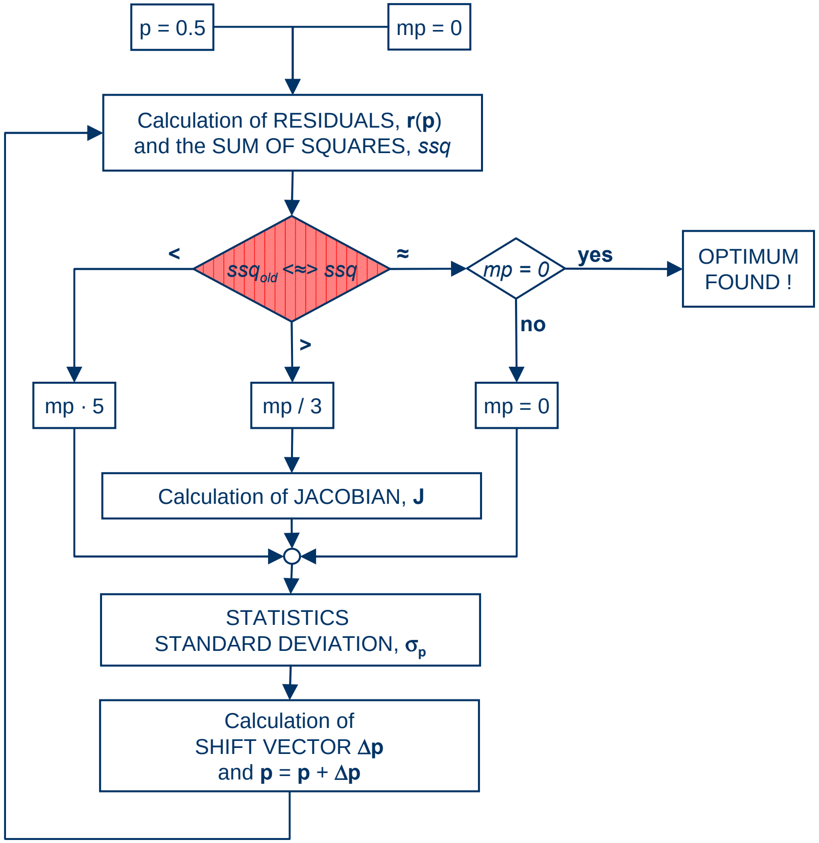
Stack: Base

Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1.6667	double
i	1	double
p	0.412	double

Command History | Workspace

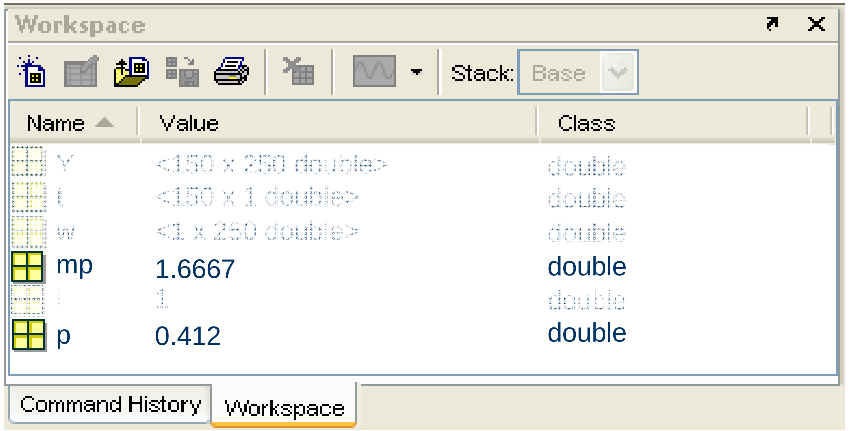


The NG/LM algorithm


p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
ssq_{old} <≈> ssq
<
≈
>
$mp = 0$
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

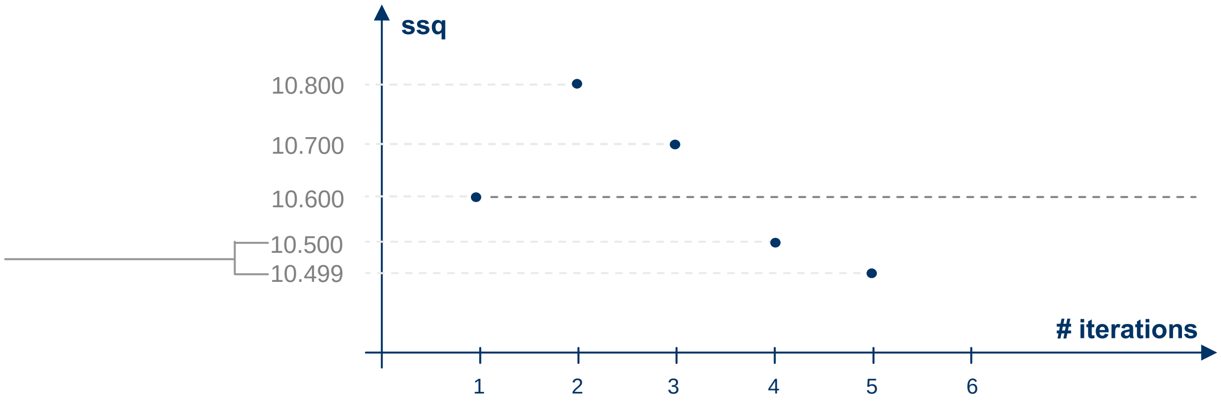
Current method :

NG/Levenberg-Marquardt method

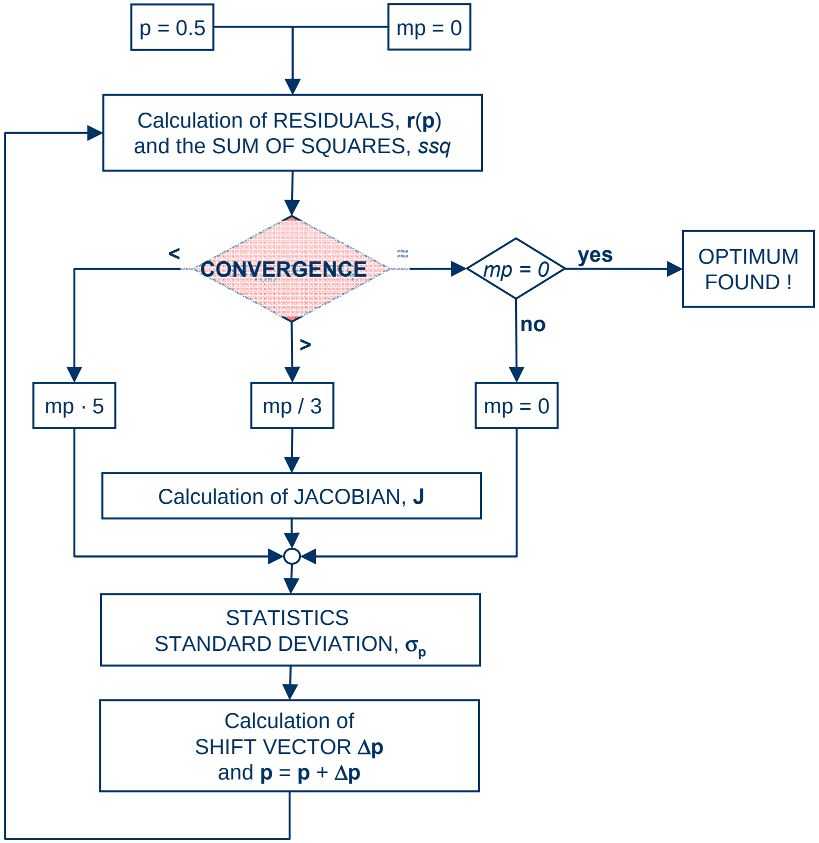


Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1.6667	double
i	1	double
p	0.412	double

$$\frac{10.500-10.499}{10.500}=9.523\cdot10^{-5}<10^{-4}$$



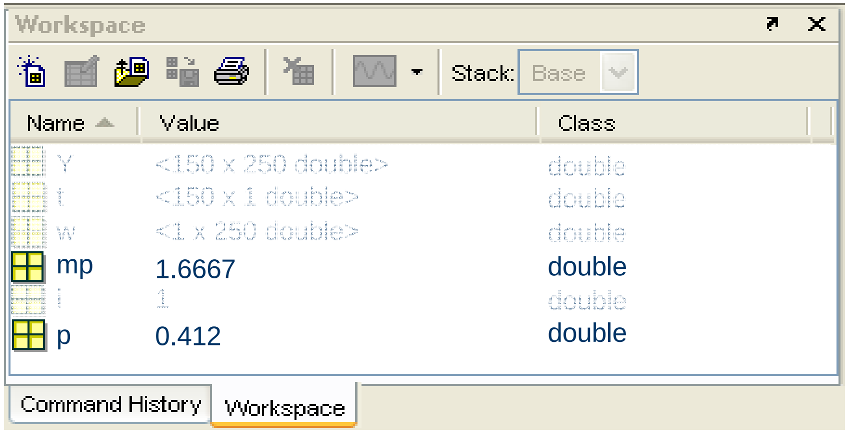
The NG/LM algorithm


p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**)
and the SUM OF SQUARES, *ssq*
CONVERGENCE
<
>
≈
mp = 0
yes
no
OPTIMUM
FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS
STANDARD DEVIATION, σ_p
Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

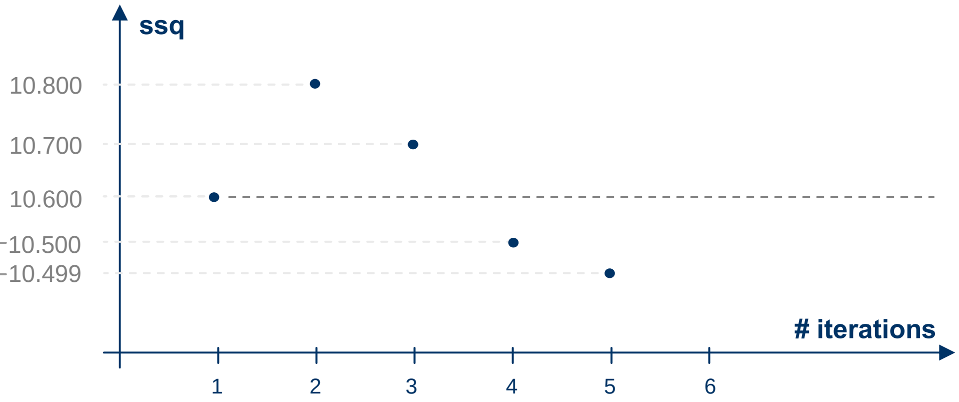
$$\frac{10.500-10.499}{10.500} = 9.523\cdot 10^{-5} < 10^{-4}$$

<u>Current method :</u>

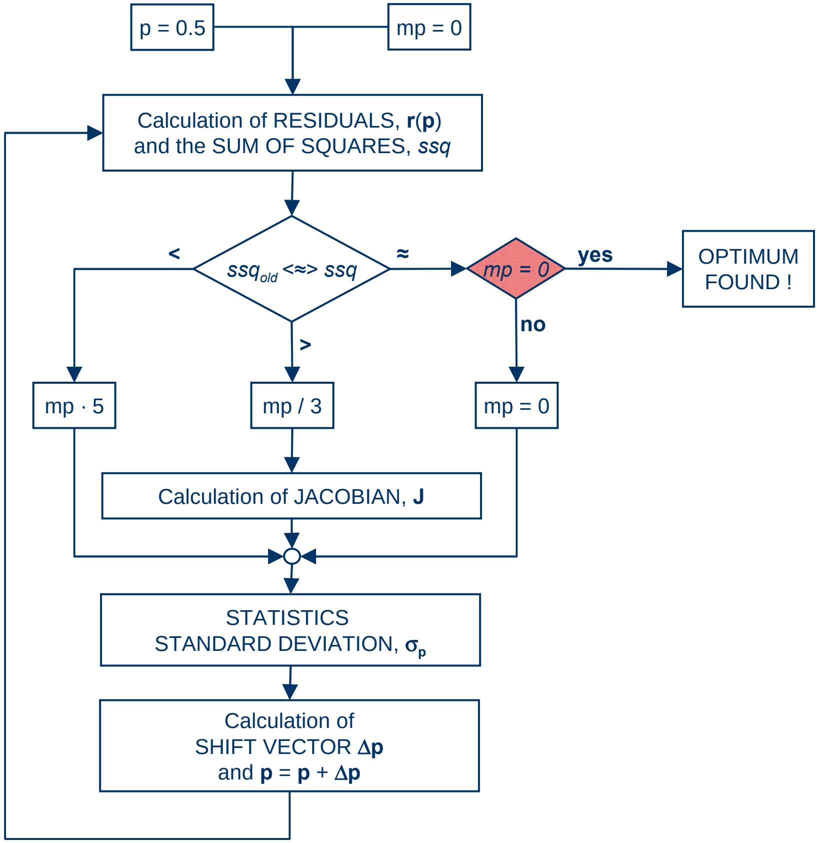
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1.6667	double
i	1	double
p	0.412	double



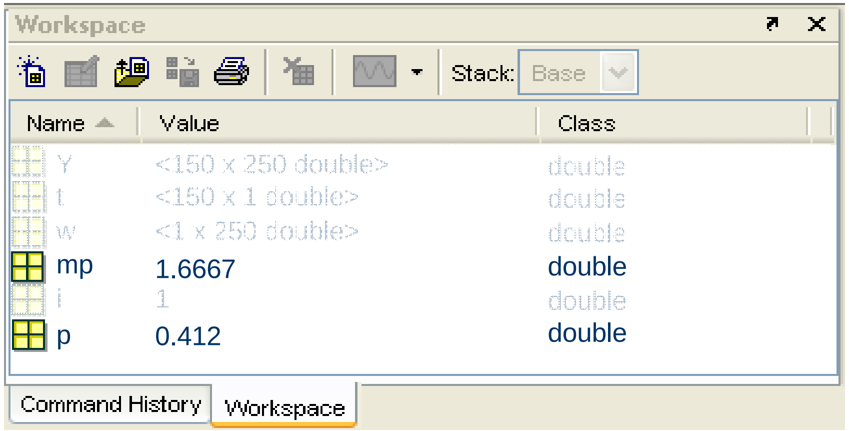
The NG/LM algorithm


p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*
<
ssq_{old} <≈> ssq
≈
mp = 0
yes
OPTIMUM FOUND !
no
>
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS STANDARD DEVIATION, σ_p
Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

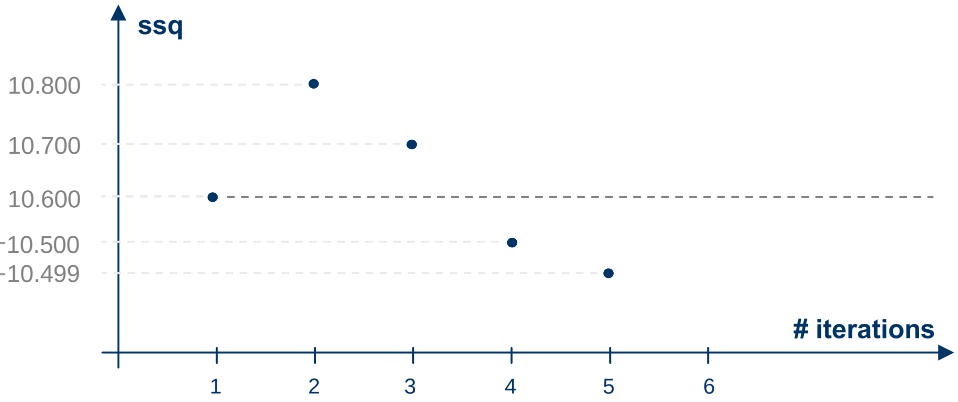
$$\frac{10.500-10.499}{10.500}=9.523\cdot10^{-5}<10^{-4}$$

__Current method :__

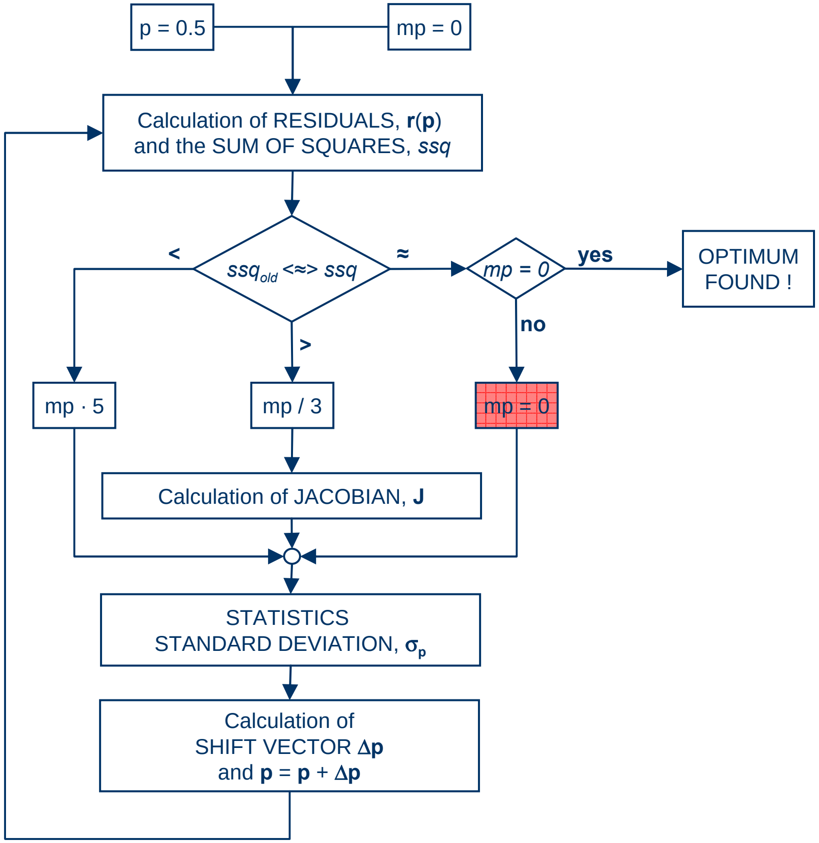
NG/Levenberg-Marquardt method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	1.6667	double
i	1	double
p	0.412	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

Calculation of JACOBIAN, **J**

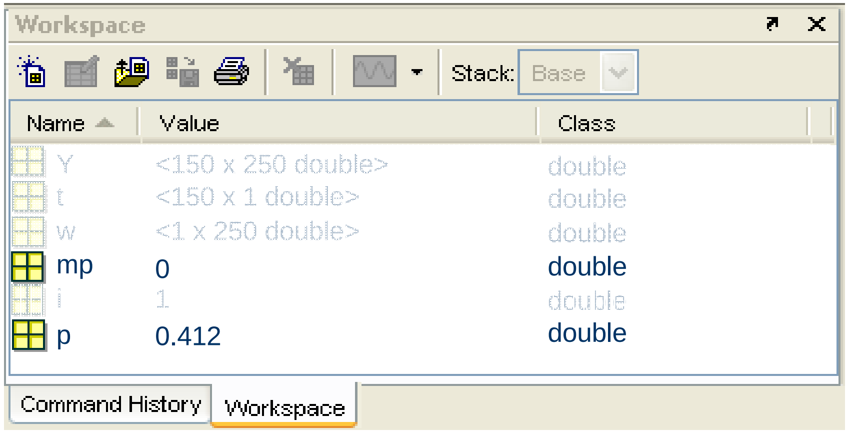
STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

$$\frac{10.500-10.499}{10.500}=9.523\cdot 10^{-5}<10^{-4}$$

<u>Current method :</u>

Newton-Gauss method

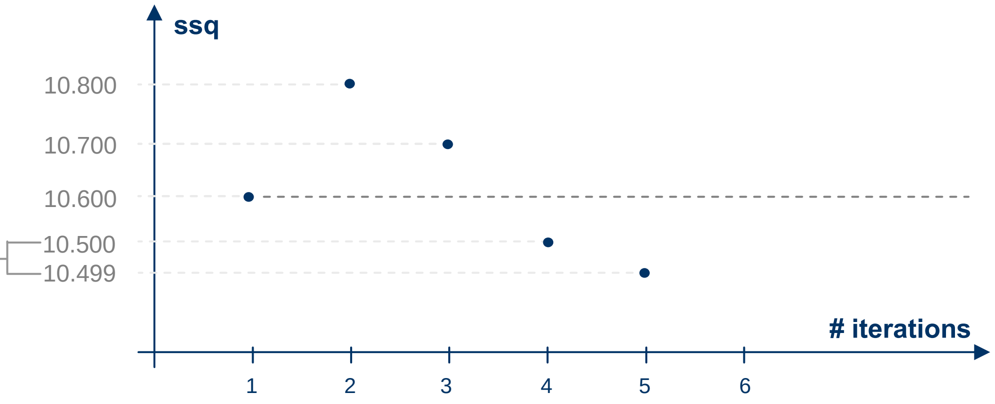


Workspace

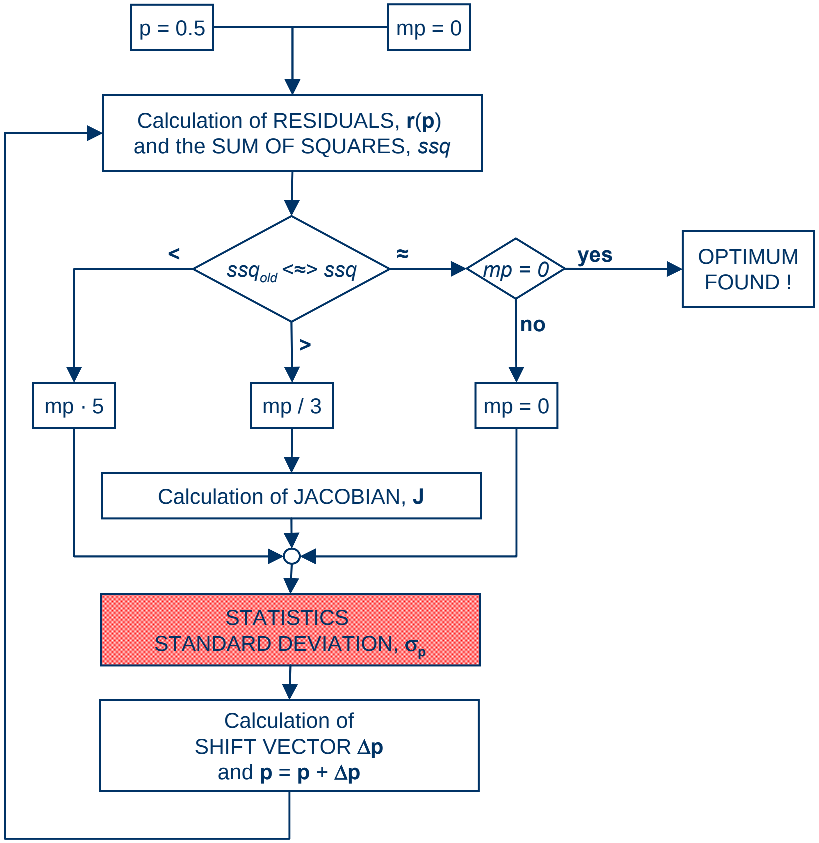
Stack: Base

Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.412	double

Command History | Workspace



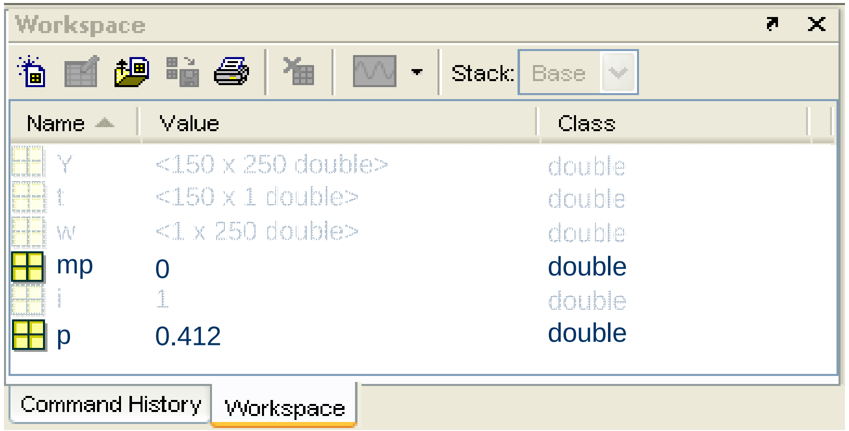
The NG/LM algorithm


p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**)
and the SUM OF SQUARES, *ssq*
<
ssq_{old} <≈> *ssq*
≈
mp = 0
yes
OPTIMUM FOUND !
>
no
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS
STANDARD DEVIATION, σ_p
Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

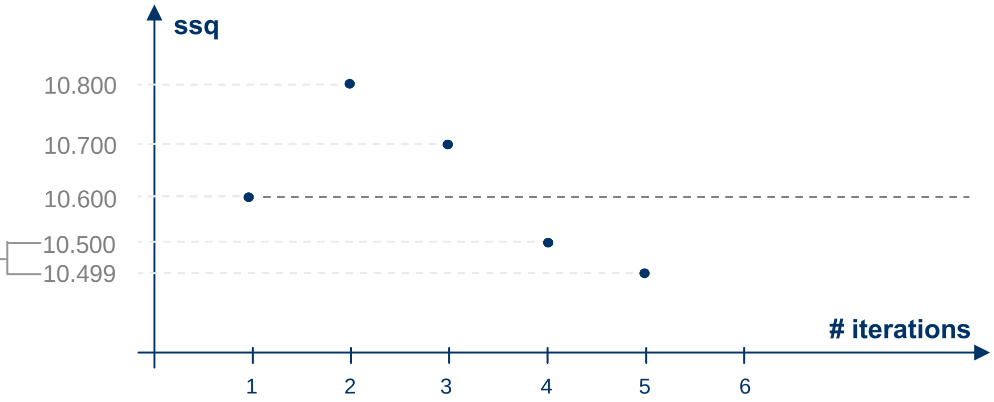
$$\frac{10.500-10.499}{10.500}=9.523\cdot 10^{-5}<10^{-4}$$

Current method :

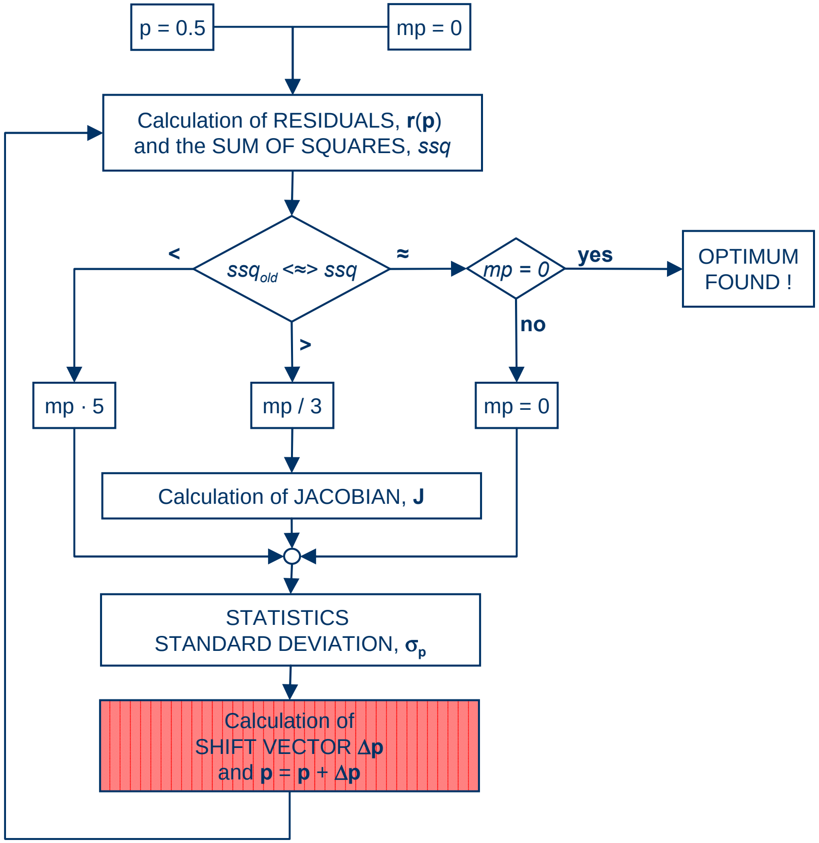
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.412	double



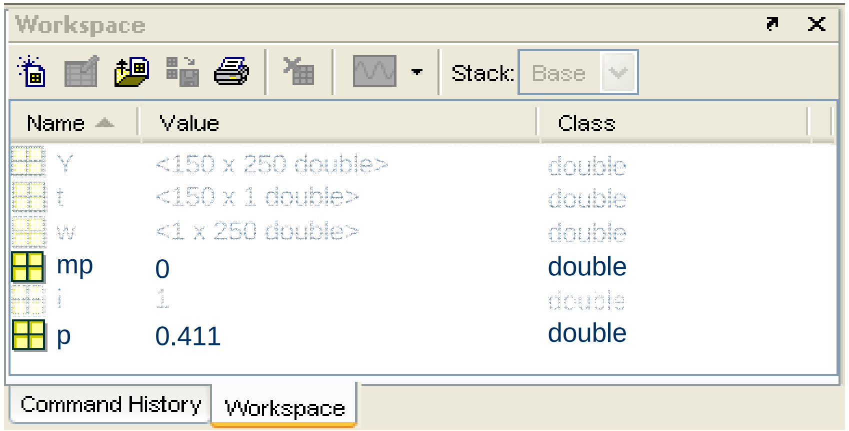
The NG/LM algorithm


p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**)
and the SUM OF SQUARES, *ssq*
<
ssq_{old} <≈> *ssq*
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS
STANDARD DEVIATION, σ_p
Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

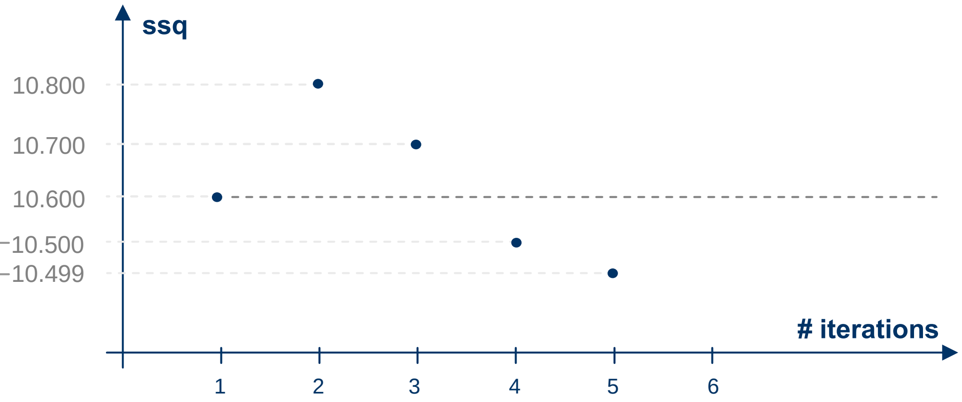
$$\frac{10.500-10.499}{10.500}=9.523\cdot10^{-5}<10^{-4}$$

Current method :

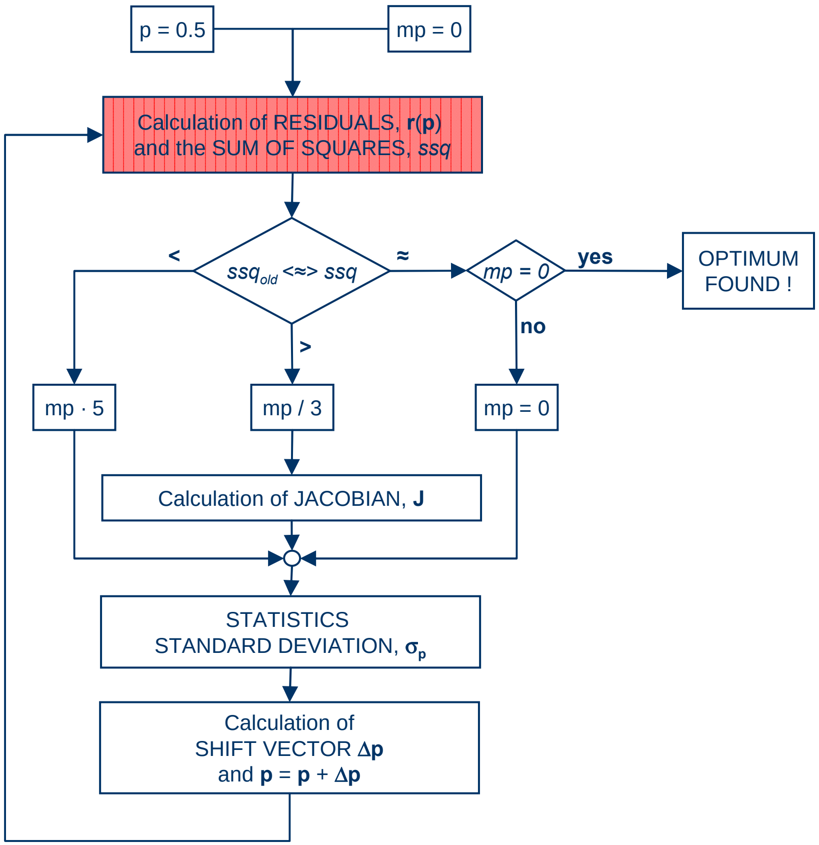
Newton-Gauss method



Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.411	double



The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r(p)** and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

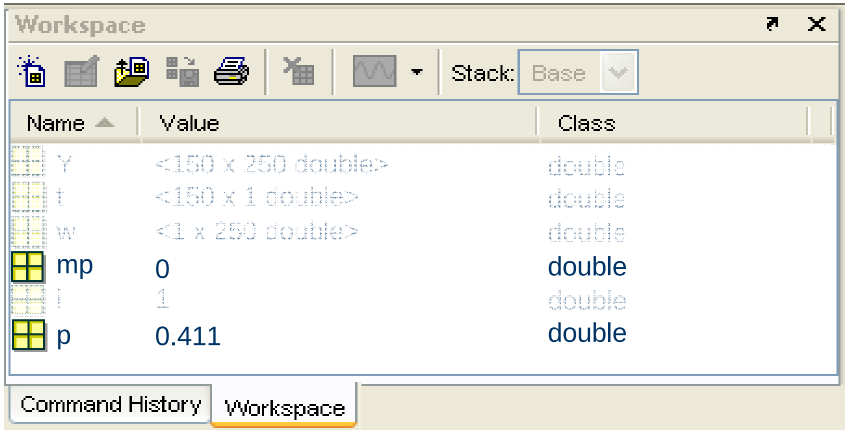
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

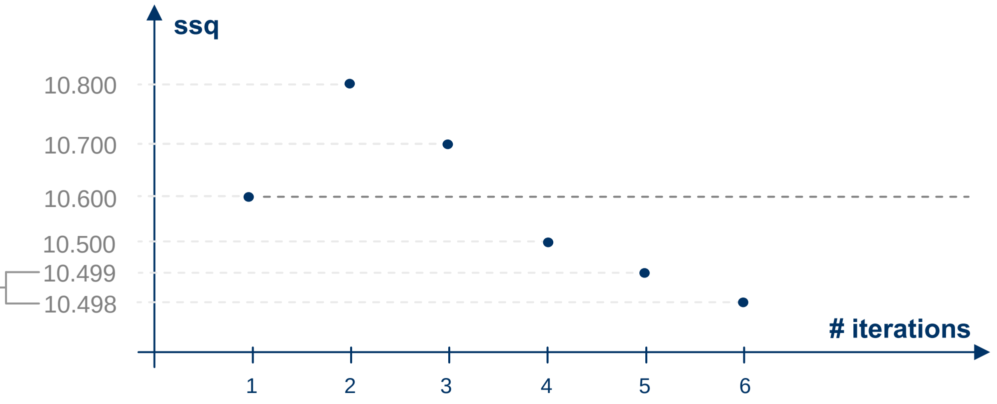
Calculation of SHIFT VECTOR **Δp** and **p** = **p** + **Δp**

Current method :

Newton-Gauss method

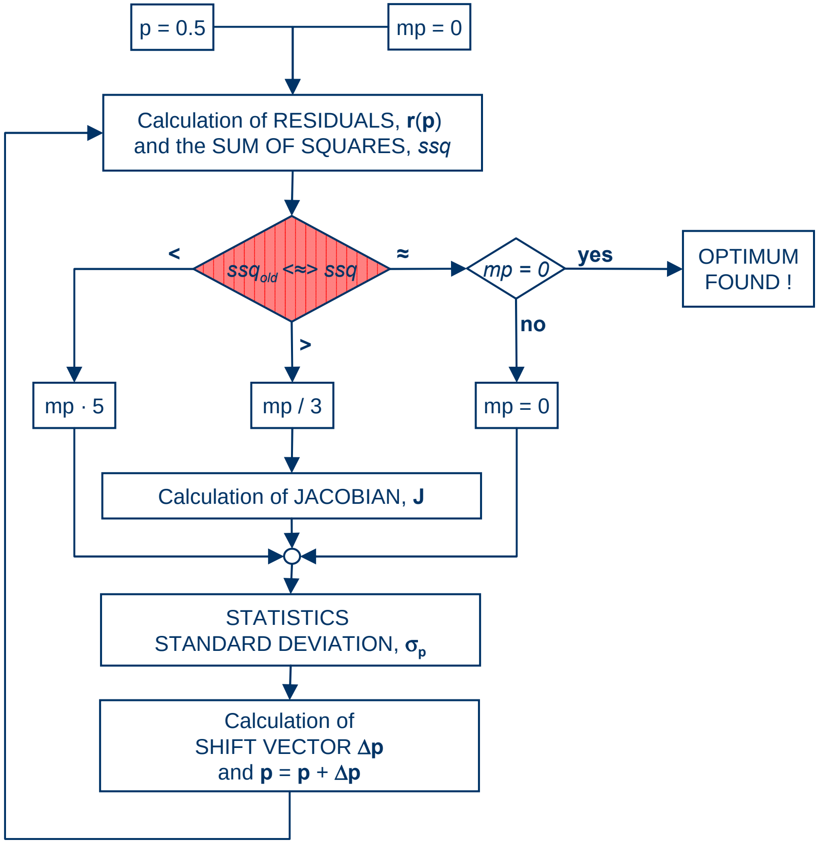


Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.411	double



$$\frac{10.499-10.498}{10.499}=9.524\cdot10^{-5}<10^{-4}$$

The NG/LM algorithm



p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> ssq

<

≈

>

$mp = 0$

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

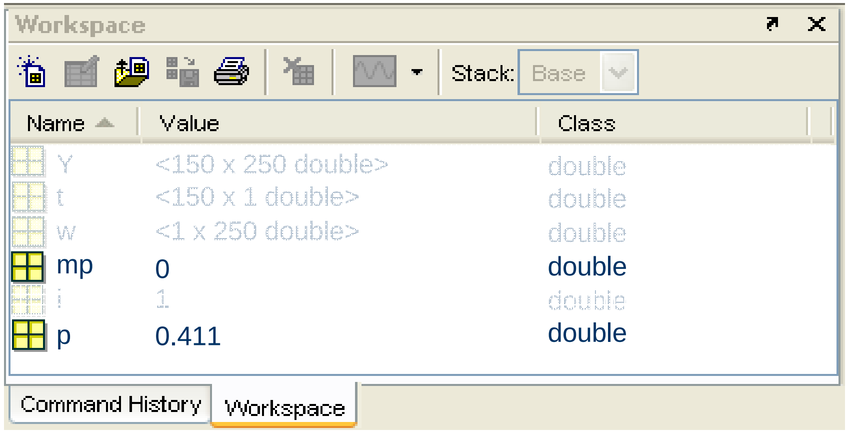
Calculation of JACOBIAN, **J**

STATISTICS
STANDARD DEVIATION, σ_p

Calculation of
SHIFT VECTOR **Δp**
and **p** = **p** + **Δp**

Current method :

Newton-Gauss method

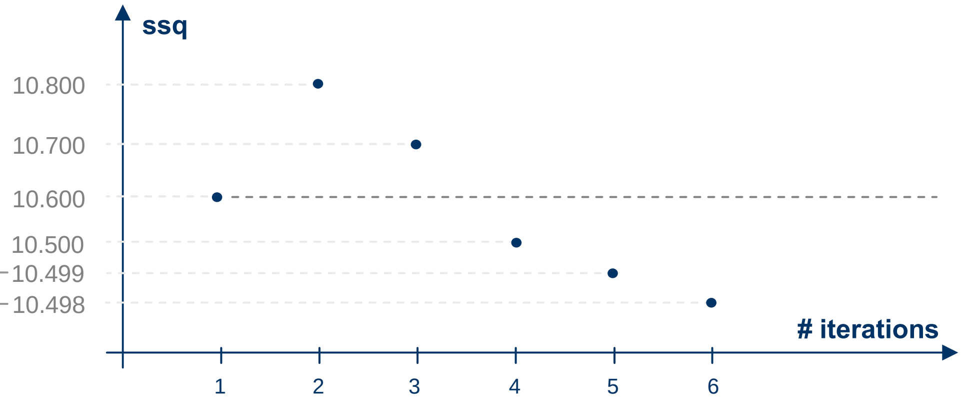


Workspace

Stack: Base

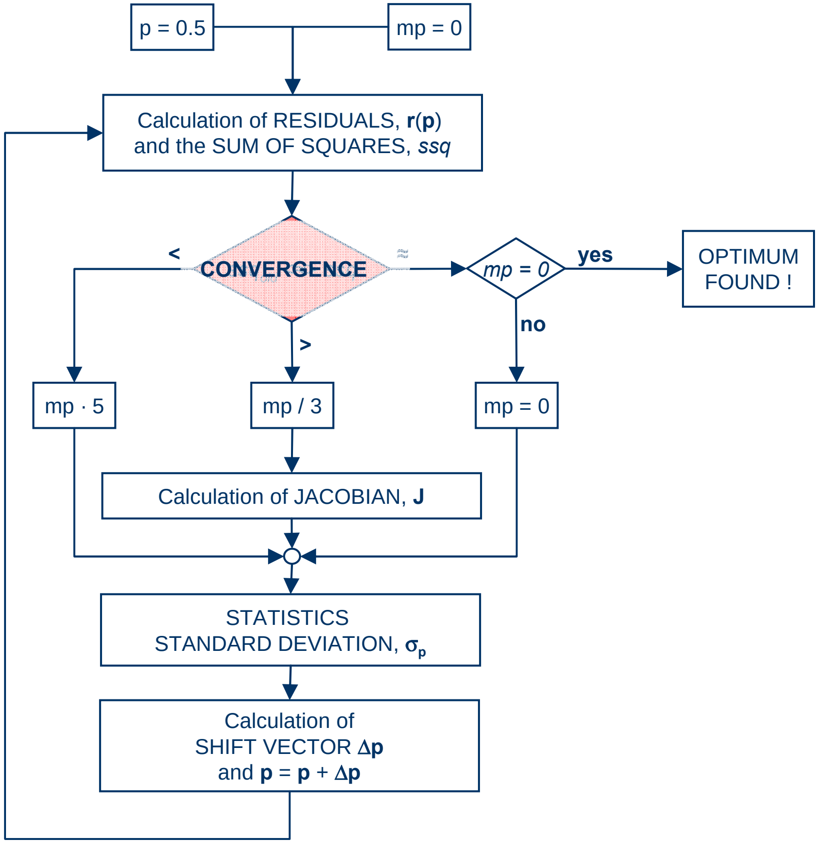
Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.411	double

Command History | Workspace



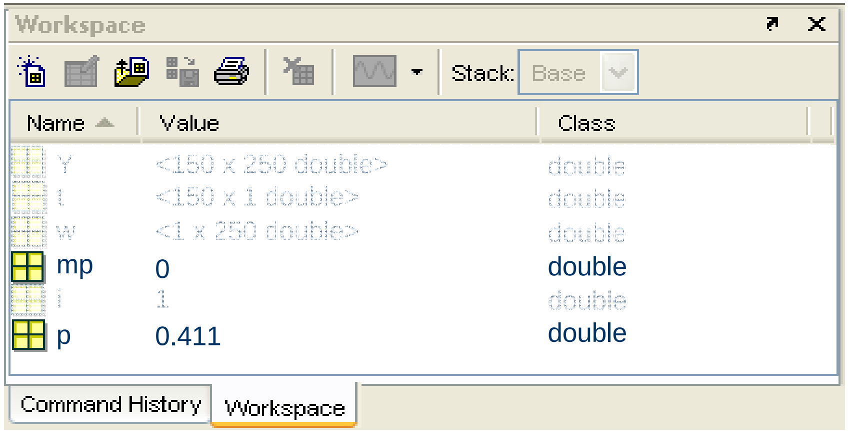
$$\frac{10.499-10.498}{10.499}=9.524\cdot10^{-5}<10^{-4}$$

The NG/LM algorithm



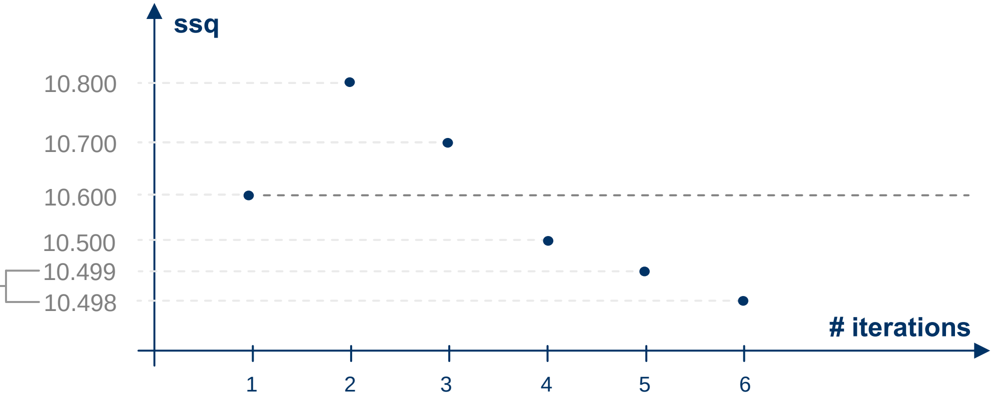
Current method :

Newton-Gauss method

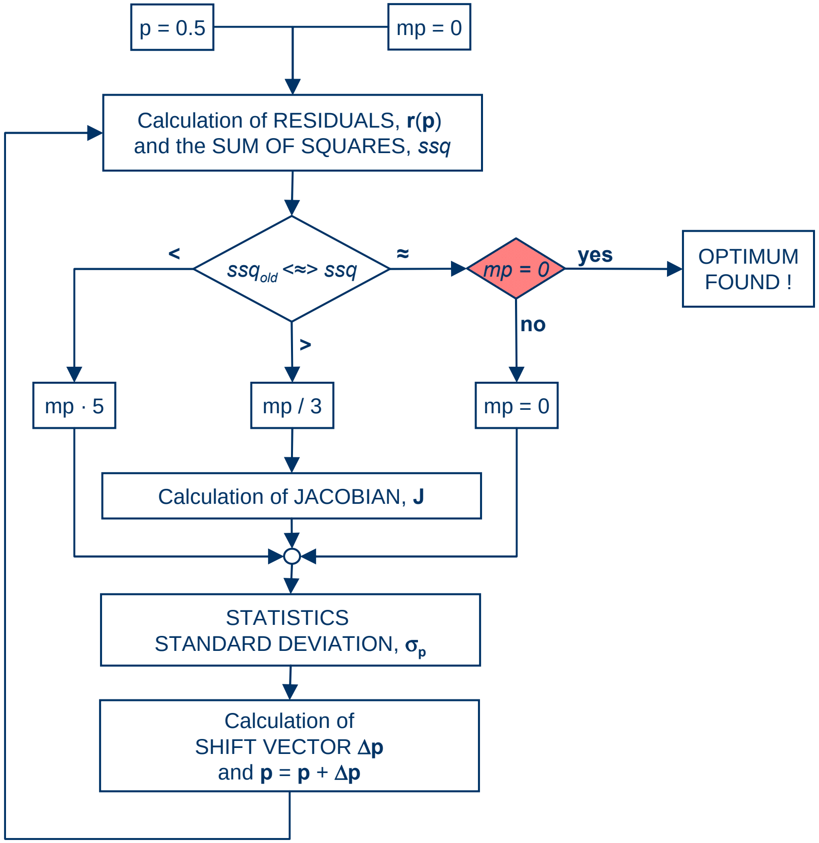


Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.411	double

$$\frac{10.499-10.498}{10.499}=9.524\cdot10^{-5}<10^{-4}$$



The NG/LM algorithm


p = 0.5

mp = 0

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

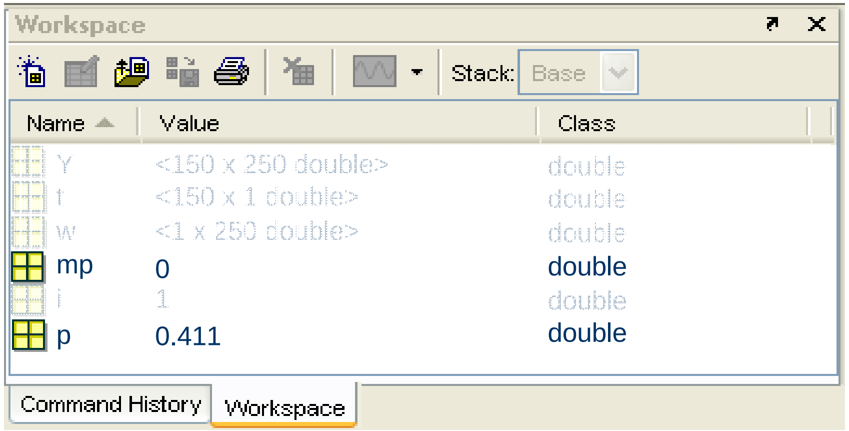
Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR **Δp** and **p** = **p** + **Δp**

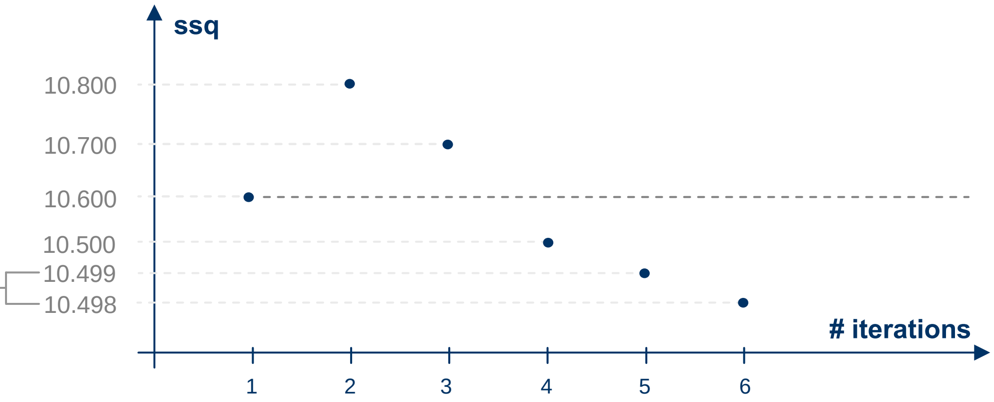
Current method :

Newton-Gauss method

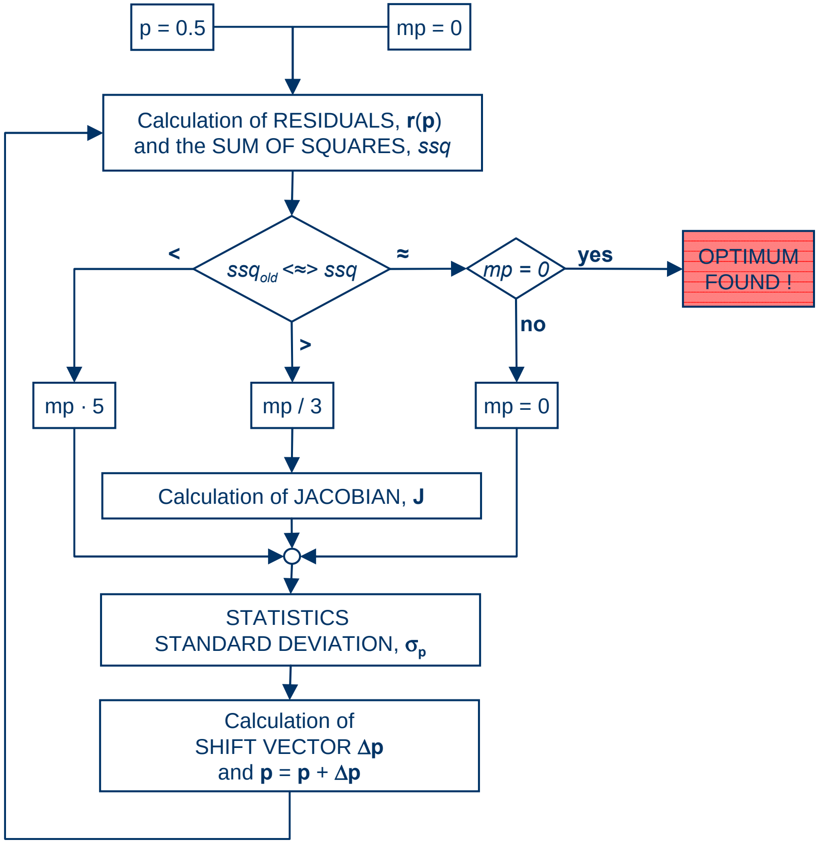


Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.411	double

$$\frac{10.499-10.498}{10.499}=9.524\cdot10^{-5}<10^{-4}$$



The NG/LM algorithm

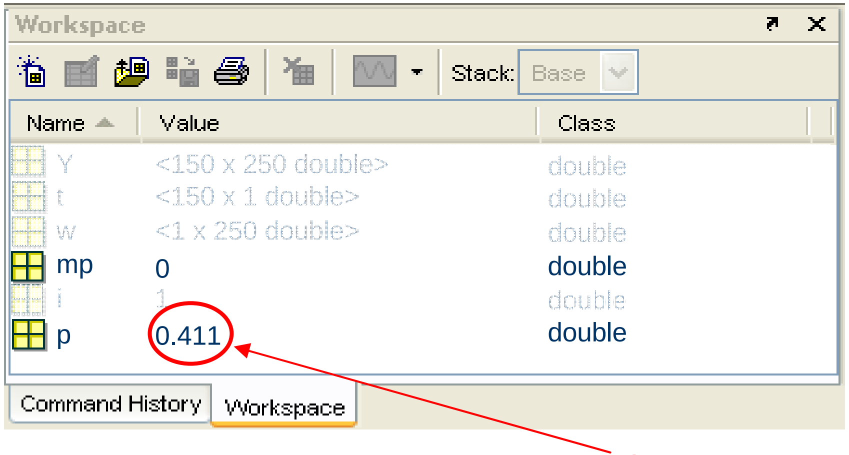


p = 0.5
mp = 0
Calculation of RESIDUALS, **r**(**p**)
and the SUM OF SQUARES, *ssq*
ssq_{old} <≈> *ssq*
<
≈
>
mp = 0
yes
no
OPTIMUM FOUND !
mp · 5
mp / 3
mp = 0
Calculation of JACOBIAN, **J**
STATISTICS
STANDARD DEVIATION, σ_p
Calculation of
SHIFT VECTOR Δ**p**
and **p** = **p** + Δ**p**

$$\frac{10.499-10.498}{10.499}=9.524\cdot10^{-5}<10^{-4}$$

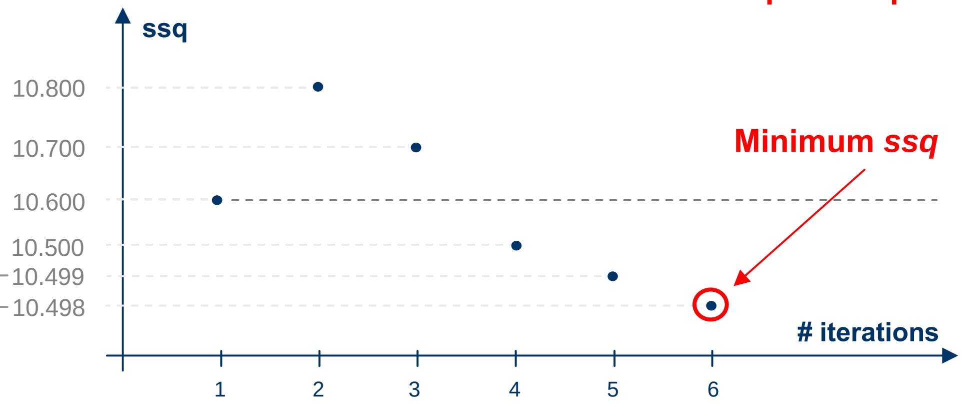
Current method :

Newton-Gauss method



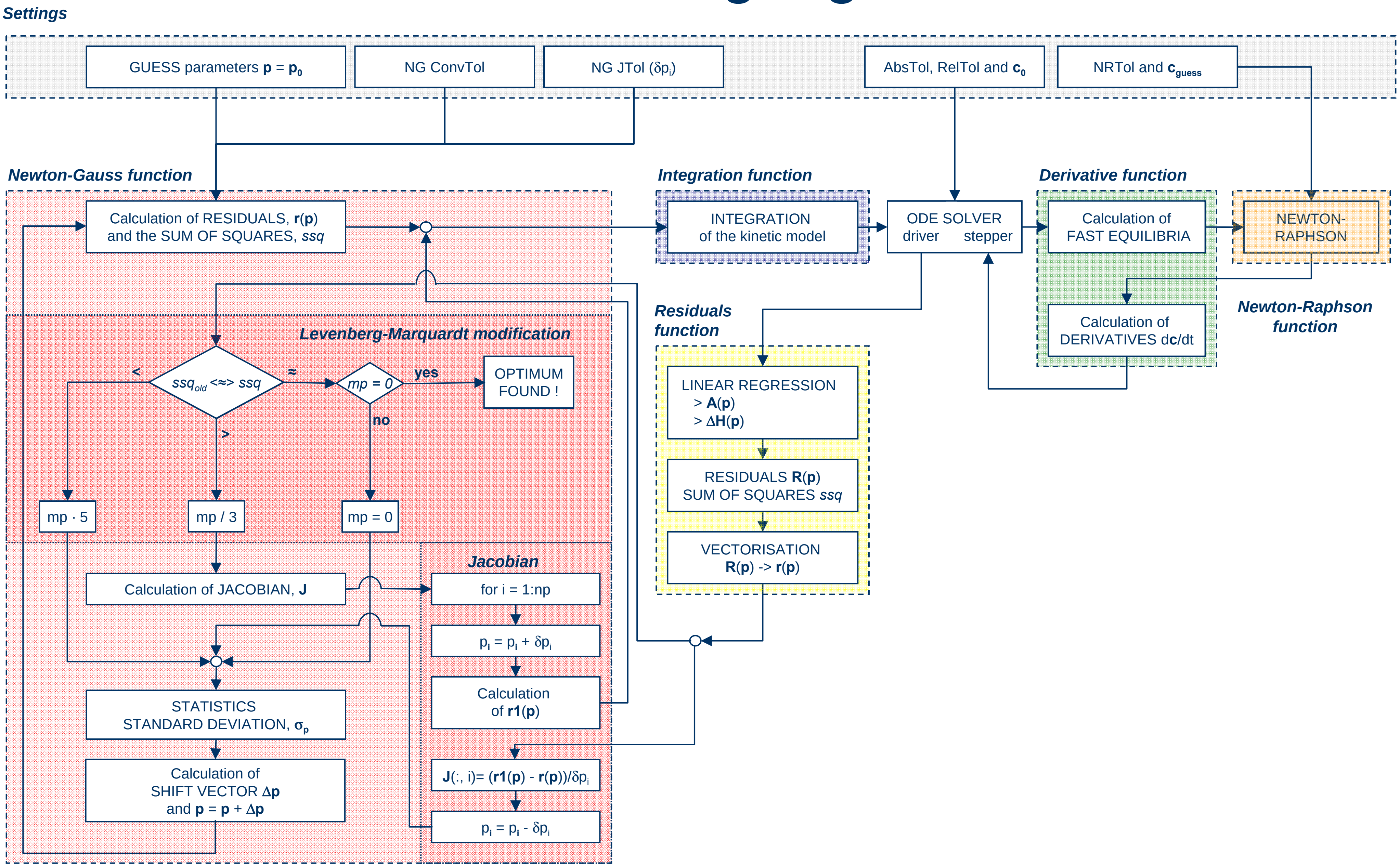
Name	Value	Class
Y	<150 x 250 double>	double
t	<150 x 1 double>	double
w	<1 x 250 double>	double
mp	0	double
i	1	double
p	0.411	double

Optimum p



Minimum *ssq*

Kinetic modeling algorithm



Settings

GUESS parameters **p** = $\mathbf{p_0}$

NG ConvTol

NG JTol (δp_i)

AbsTol, RelTol and $\mathbf{c_0}$

NRTol and $\mathbf{c}_{guess}$

Newton-Gauss function

Calculation of RESIDUALS, **r**(**p**) and the SUM OF SQUARES, *ssq*

Levenberg-Marquardt modification

ssq_{old} <≈> *ssq*

<

≈

>

mp = 0

yes

no

OPTIMUM FOUND !

mp · 5

mp / 3

mp = 0

Calculation of JACOBIAN, **J**

STATISTICS STANDARD DEVIATION, σ_p

Calculation of SHIFT VECTOR Δ**p** and **p** = **p** + Δ**p**

Jacobian

for i = 1:np

$p_i = p_i + \delta p_i$

Calculation of **r1**(**p**)

J(:, i)= (**r1**(**p**) - **r**(**p**))/δp_i

$p_i = p_i - \delta p_i$

Integration function

INTEGRATION of the kinetic model

ODE SOLVER driver stepper

Residuals function

LINEAR REGRESSION
> **A**(**p**)
> Δ**H**(**p**)

RESIDUALS **R**(**p**) SUM OF SQUARES *ssq*

VECTORISATION **R**(**p**) -> **r**(**p**)

Derivative function

Calculation of FAST EQUILIBRIA

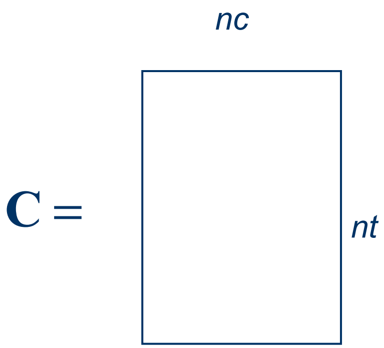
Calculation of DERIVATIVES d**c**/dt

NEWTON-RAPHSON

Newton-Raphson function

Rank Deficiency

of the concentration profile



The maximum *rank* of **C** is :

$$\boxed{rank_{max}\left(\mathbf{C}\right)=\min\left(nt,nc\right)}$$

Rank Deficiency

of the concentration profile

$\mathbf{C} =$

The maximum *rank* of **C** is :

$$\boxed{rank_{max}\left(\mathbf{C}\right) = \min\left(nt, nc\right)}$$

The pseudo inverse $\left(\mathbf{C}^{+}\right)$ only exists if : $rank\left(\mathbf{C}\right) = rank_{max}\left(\mathbf{C}\right)$

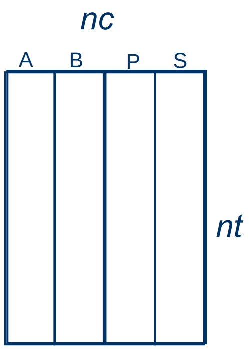
$\Leftrightarrow$

$rank_{max}\left(\mathbf{C}\right)$ columns or rows are linearly independant

Rank Deficiency

of the concentration profile

$\mathbf{C} =$



The maximum *rank* of **C** is :

$$\boxed{rank_{max}\left(\mathbf{C}\right) = \min\left(nt, nc\right)}$$

The pseudo inverse $\left(\mathbf{C}^{+}\right)$ only exists if : $rank\left(\mathbf{C}\right) = rank_{max}\left(\mathbf{C}\right)$

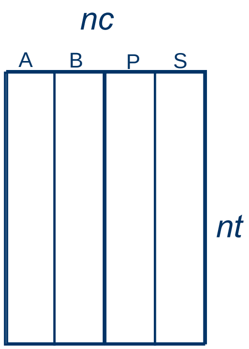
$\Leftrightarrow$

$rank_{max}\left(\mathbf{C}\right)$ columns or rows are linearly independant

Example : $A + B \xrightarrow{k} P + S \quad \left(in\ batch\ conditions\right)$

Rank Deficiency
of the concentration profile

$\mathbf{C} =$



The maximum *rank* of **C** is :

$$\boxed{rank_{max}\left(\mathbf{C}\right) = \min\left(nt, nc\right)}$$

The pseudo inverse $\left(\mathbf{C}^{+}\right)$ only exists if : $rank\left(\mathbf{C}\right) = rank_{max}\left(\mathbf{C}\right)$

$\Leftrightarrow$

$rank_{max}\left(\mathbf{C}\right)$ columns or rows are linearly independant

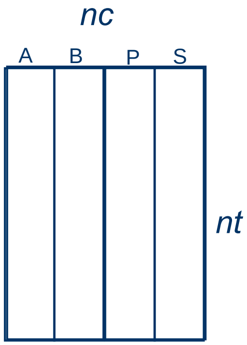
<u>Example</u> : $A + B \xrightarrow{k} P + S \quad \left(in\ batch\ conditions\right)$

- *Maximum possible rank ?*
- *Number of independant species in stoichiometric conditions (A_0 = B_0 = 1) ?*
- *And in non-stoichiometric conditions (A_0 = 1, B_0 = 0.5) ?*

Rank Deficiency

of the concentration profile

$\mathbf{C} =$



The maximum *rank* of **C** is :

$$\boxed{rank_{max}\left(\mathbf{C}\right) = \min\left(nt, nc\right)}$$

The pseudo inverse $\left(\mathbf{C}^{+}\right)$ only exists if : $rank\left(\mathbf{C}\right) = rank_{max}\left(\mathbf{C}\right)$

$\Leftrightarrow$

$rank_{max}\left(\mathbf{C}\right)$ columns or rows are linearly independant

Example : $A + B \xrightarrow{k} P + S \quad \left(in\ batch\ conditions\right)$

- *Maximum possible rank ?*

$$4\ species \;\Rightarrow\; rank_{max}\left(\mathbf{C}\right) = 4$$

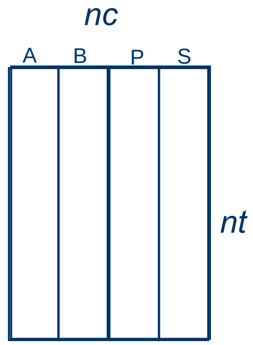
- *Number of independant species in stoichiometric conditions ($A_0 = B_0 = 1$) ?*

- *And in non-stoichiometric conditions ($A_0 = 1$, $B_0 = 0.5$) ?*

Rank Deficiency

of the concentration profile

$\mathbf{C} =$



The maximum *rank* of **C** is :

$$\boxed{rank_{max}\left(\mathbf{C}\right) = \min\left(nt, nc\right)}$$

The pseudo inverse $\left(\mathbf{C}^{+}\right)$ only exists if : $rank\left(\mathbf{C}\right) = rank_{max}\left(\mathbf{C}\right)$

$\Leftrightarrow$

$rank_{max}\left(\mathbf{C}\right)$ columns or rows are linearly independant

Example : $A + B \xrightarrow{k} P + S \quad \left(in\ batch\ conditions\right)$

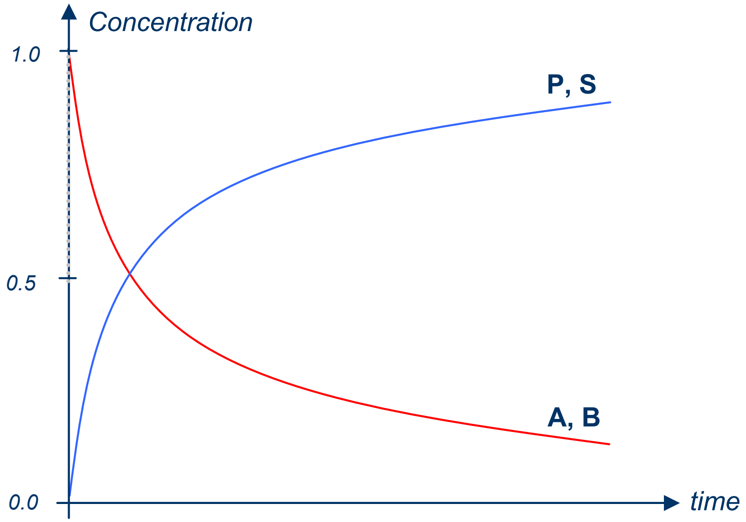
- *Maximum possible rank ?*

$$4\ species \ \Rightarrow \ rank_{max}\left(\mathbf{C}\right) = 4$$

- *Number of independant species in stoichiometric conditions ($A_0 = B_0 = 1$) ?*

$$rank\left(\mathbf{C}\right) = 2 \quad (only!)$$

- *And in non-stoichiometric conditions ($A_0 = 1$, $B_0 = 0.5$) ?*

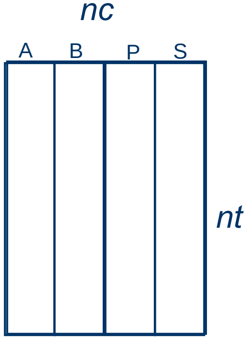


***[A]$_t$** and **[B]$_t$** are identical*
***[P]$_t$** and **[S]$_t$** are identical*

1 species
1 species
→ *2 independant species*

Rank Deficiency
of the concentration profile

$\mathbf{C} =$



The maximum *rank* of **C** is :

$$\boxed{rank_{max}\left(\mathbf{C}\right) = \min\left(nt, nc\right)}$$

The pseudo inverse $\left(\mathbf{C}^{+}\right)$ only exists if : $rank\left(\mathbf{C}\right) = rank_{max}\left(\mathbf{C}\right)$

$\Leftrightarrow$

$rank_{max}\left(\mathbf{C}\right)$ columns or rows are linearly independant

Example : $A + B \xrightarrow{k} P + S \quad \left(in\ batch\ conditions\right)$

- *Maximum possible rank ?*

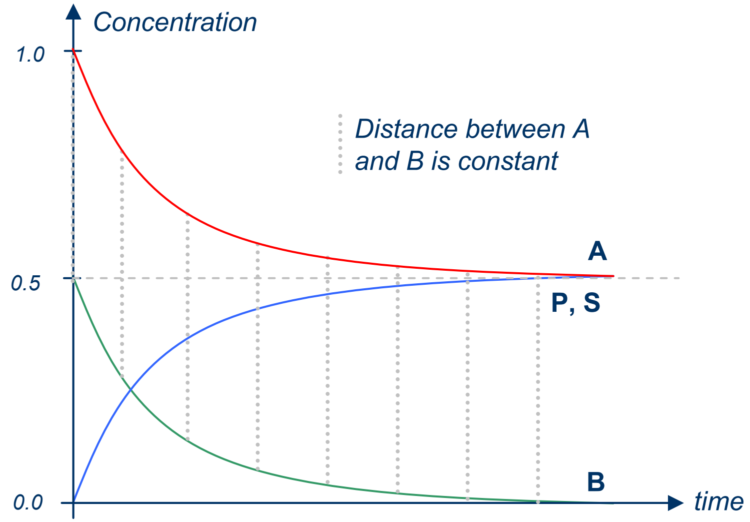
$$4\ species \;\Rightarrow\; rank_{max}\left(\mathbf{C}\right) = 4$$

- *Number of independant species in stoichiometric conditions ($A_0 = B_0 = 1$) ?*

$$rank\left(\mathbf{C}\right) = 2 \quad (only!)$$

- *And in non-stoichiometric conditions ($A_0 = 1$, $B_0 = 0.5$) ?*

$$rank\left(\mathbf{C}\right) = 2 \quad (still!)$$



$[A]_t$ and $[B]_t$ disappear at the same rate — *1 species*
$[P]_t$ and $[S]_t$ are identical — *1 species*
→ *2 independant species*

Annihilation of Rank Deficiency

5 ways to break the rank deficiency :

Annihilation of Rank Deficiency

5 ways to break the rank deficiency :

- **Model Reduction** : set the dependant species as colorless (non-absorbing)

⇒ *Rates constants will be correct*
Absorption spectra will be wrong (mixed pure spectra)

Annihilation of Rank Deficiency

5 ways to break the rank deficiency :

- **Model Reduction** : set the dependant species as colorless (non-absorbing)

  ⇒ *Rates constants will be correct*
  Absorption spectra will be wrong (mixed pure spectra)

- Provide **Known Spectra**

Annihilation of Rank Deficiency

5 ways to break the rank deficiency :

- **Model Reduction** : set the dependant species as colorless (non-absorbing)

  ⇒ *Rates constants will be correct*
  Absorption spectra will be wrong (mixed pure spectra)

- Provide **Known Spectra**

- Work under **Semi-Batch Conditions**

Annihilation of Rank Deficiency

5 ways to break the rank deficiency :

- **Model Reduction** : set the dependant species as colorless (non-absorbing)

  ⇒ *Rates constants will be correct*
  Absorption spectra will be wrong (mixed pure spectra)

- Provide **Known Spectra**

- Work under **Semi-Batch Conditions**

- Use concentration dependant measurements
  (**Second Order Global Analysis**)

Annihilation of Rank Deficiency

5 ways to break the rank deficiency :

- **Model Reduction** : set the dependant species as colorless (non-absorbing)

  ⇒ *Rates constants will be correct*
  Absorption spectra will be wrong (mixed pure spectra)

- Provide **Known Spectra**

- Work under **Semi-Batch Conditions**

- Use concentration dependant measurements
  (**Second Order Global Analysis**)

- Extend the wavelength-time domain to resolve linear dependencies
  (**Tri-linear Measurements**)

  Example : Coupling chromatrography to UV

Annihilation of Rank Deficiency

5 ways to break the rank deficiency :

- **Model Reduction** : set the dependant species as colorless (non-absorbing)

  ⇒ *Rates constants will be correct*
  Absorption spectra will be wrong (mixed pure spectra)

- Provide **Known Spectra**

- Work under **Semi-Batch Conditions**

- Use concentration dependant measurements
  (**Second Order Global Analysis**)

- Extend the wavelength-time domain to resolve linear dependencies
  (**Tri-linear Measurements**)

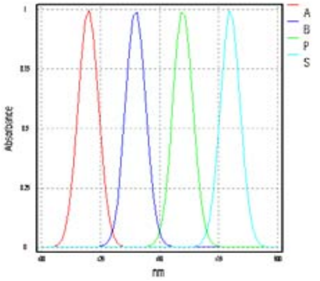
  Example : Coupling Chromatography to UV

This last method is not addressed here,
because it is highly complex

Reduced model

$$A(1)+B(0.5)\xrightarrow{k=0.5}P+S$$

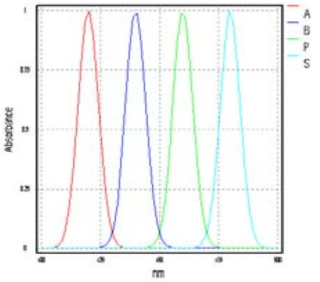
Simulated absorption spectra



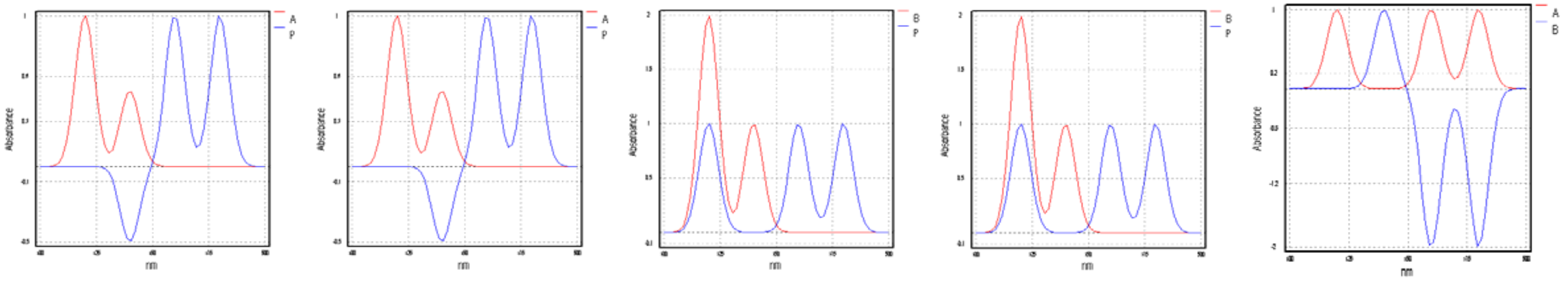
Reduced model

$$A(1)+B(0.5)\xrightarrow{k=0.5}P+S$$

Simulated absorption spectra



Fitted absorption spectra (mixed)

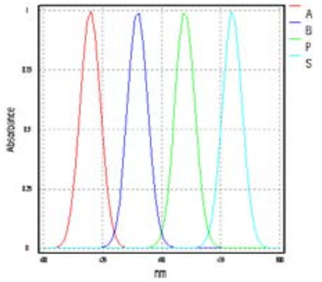


Colored species :	***A, S***	***A, P***	***B, P***	***B, S***	***A, B***
	$k_{fitted} = 0.5$	$k_{fitted} = 0.5$	$k_{fitted} = 0.5$	$k_{fitted} = 0.5$	$k_{fitted} = 0.5$

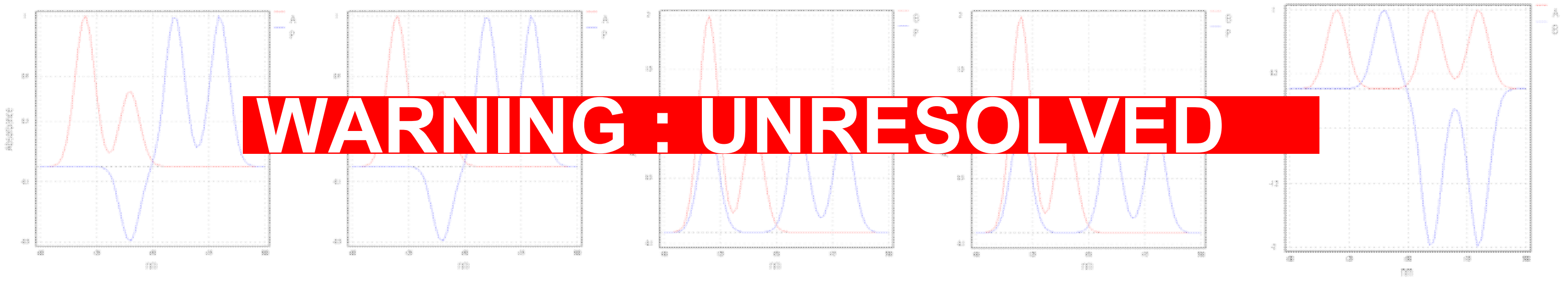
Reduced model

$$A(1)+B(0.5)\xrightarrow{k=0.5}P+S$$

Simulated absorption spectra



Fitted absorption spectra (mixed)



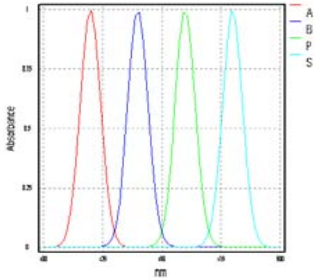
WARNING : UNRESOLVED

Colored species :	***A, S***	***A, P***	***B, P***	***B, S***	***A, B***
	$k_{fitted} = 0.5$	$k_{fitted} = 0.5$	$k_{fitted} = 0.5$	$k_{fitted} = 0.5$	$k_{fitted} = 0.5$

Add known spectra

$$A(1)+B(0.5)\xrightarrow{k=0.5}P+S$$

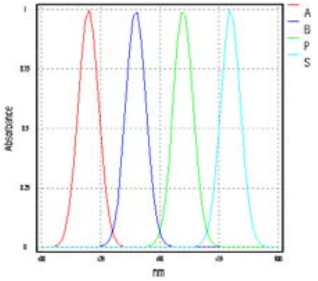
Simulated absorption spectra



Add known spectra

Simulated absorption spectra

$$A(1)+B(0.5)\xrightarrow{k=0.5}P+S$$

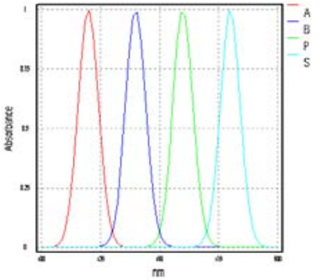


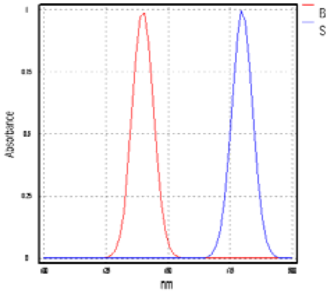
2 species are dependent

Add known spectra

Simulated absorption spectra

$$A(1)+B(0.5)\xrightarrow{k=0.5}P+S$$



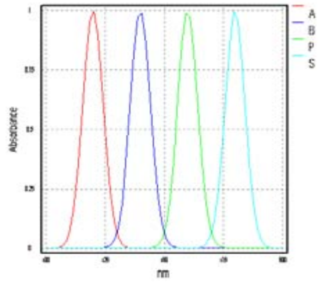


2 species are dependent

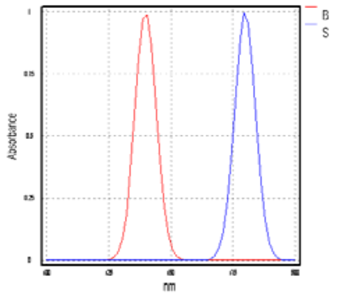
Let's provide 2 pure spectra : those of B and S

Add known spectra

Simulated absorption spectra



$$A(1)+B(0.5)\xrightarrow{k=0.5}P+S$$



2 species are dependent

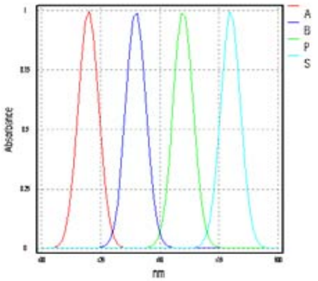
Let’s provide 2 pure spectra : those of B and S

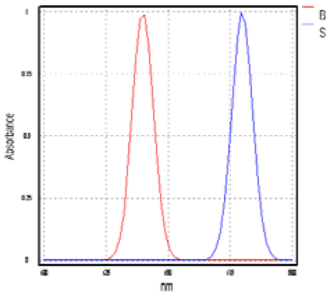
All species are set colored

Add known spectra

Simulated absorption spectra

$$A(1)+B(0.5)\xrightarrow{k=0.5}P+S$$



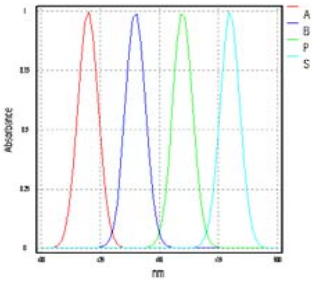


2 species are dependent

Let's provide 2 pure spectra : those of B and S

All species are set colored

Fitted absorption spectra :

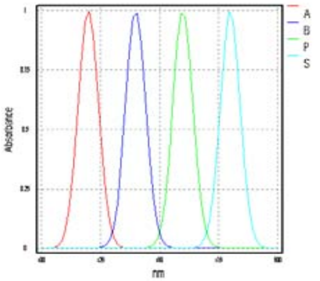


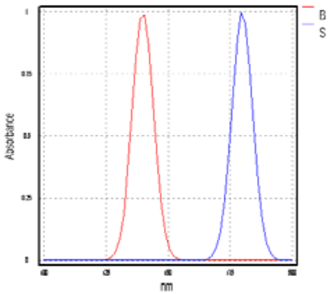
k_{fitted} = 0.5

Add known spectra

Simulated absorption spectra

$$A(1) + B(0.5) \xrightarrow{k=0.5} P + S$$



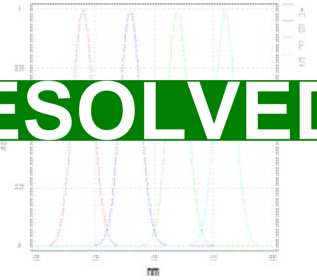


2 species are dependent

Let's provide 2 pure spectra : those of B and S

All species are set colored

Fitted absorption spectra :



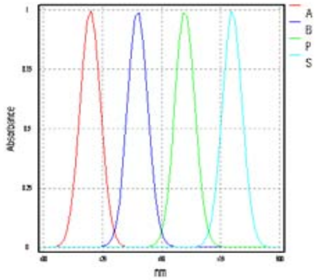
RESOLVED

k**$_{fitted}$ **= 0.5

Work under semi-batch conditions

$$A + B \xrightarrow{k=0.5} P + S$$

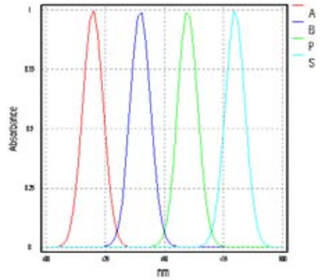
Simulated absorption spectra



Work under semi-batch conditions

Simulated absorption spectra

$$A + B \xrightarrow{k=0.5} P + S$$

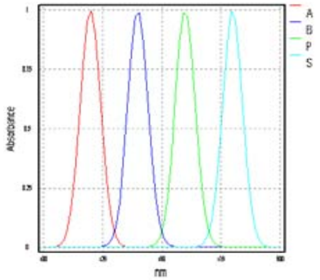


Dose A into B will break the rank deficiency between A and B but not between P and S !!!

Work under semi-batch conditions

$$A + B \xrightarrow{k=0.5} P + S$$

Simulated absorption spectra

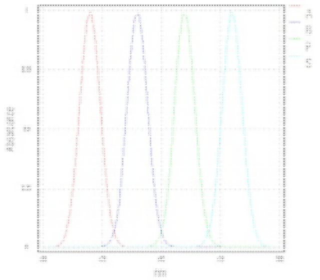


Dose A into B will break the rank deficiency between A and B but not between P and S !!!

To break both rank deficiencies, one has to dose :

- *A and P into B or*
- *A and S into B or*
- *B and P into A or*
- *B and S into A*

Fitted absorption spectra :



k_{fitted} = 0.5

Work under semi-batch conditions

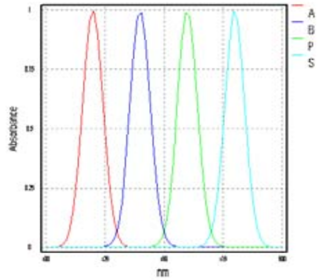
$$A + B \xrightarrow{k=0.5} P + S$$

Dose A into B will break the rank deficiency between A and B but not between P and S !!!

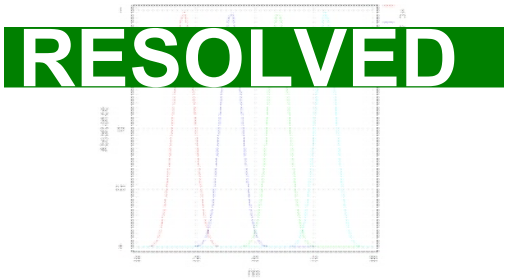
To break both rank deficiencies, one has to dose :

- *A and P into B or*
- *A and S into B or*
- *B and P into A or*
- *B and S into A*

Simulated absorption spectra



Fitted absorption spectra :

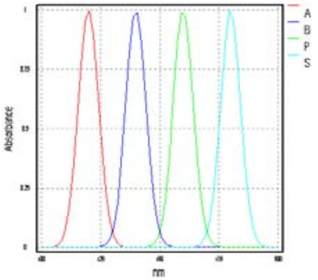

RESOLVED

k_{fitted} = 0.5

Work under semi-batch conditions

$$A + B \xrightarrow{k=0.5} P + S$$

Simulated absorption spectra

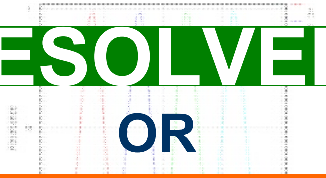


Dose A into B will break the rank deficiency between A and B but not between P and S !!!

To break both rank deficiencies, one has to dose :

- *A and P into B or*
- *A and S into B or*
- *B and P into A or*
- *B and S into A*

Fitted absorption spectra :



RESOLVED

OR

PARTIALLY RESOLVED

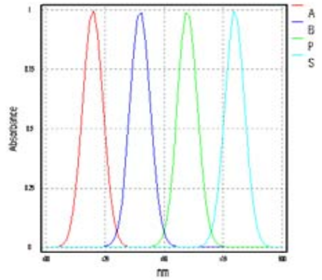
k_{fitted} = 0.5

NB : in practice, one dose A in B or B in A and set one of the two products (P or S) as uncolored. In such case, one spectrum of the two products is unresolved

Second Order Global Analysis

$$A + B \xrightarrow{k=0.5} P + S$$

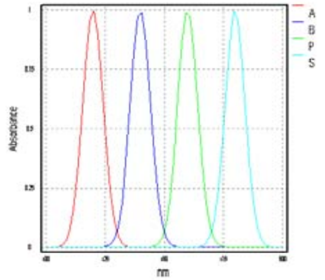
Simulated absorption spectra



Second Order Global Analysis

$$A+B\xrightarrow{k=0.5}P+S$$

Simulated absorption spectra



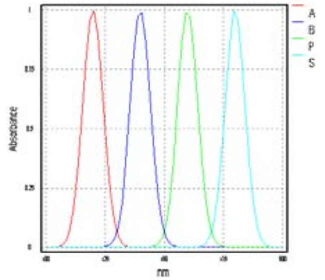
Let’s make 3 Concentration dependent measurements

	A_0	B_0
# 1	*1*	*1*
# 2	*0.5*	*1*
# 3	*1*	*0.5*

Second Order Global Analysis

$$A + B \xrightarrow{k=0.5} P + S$$

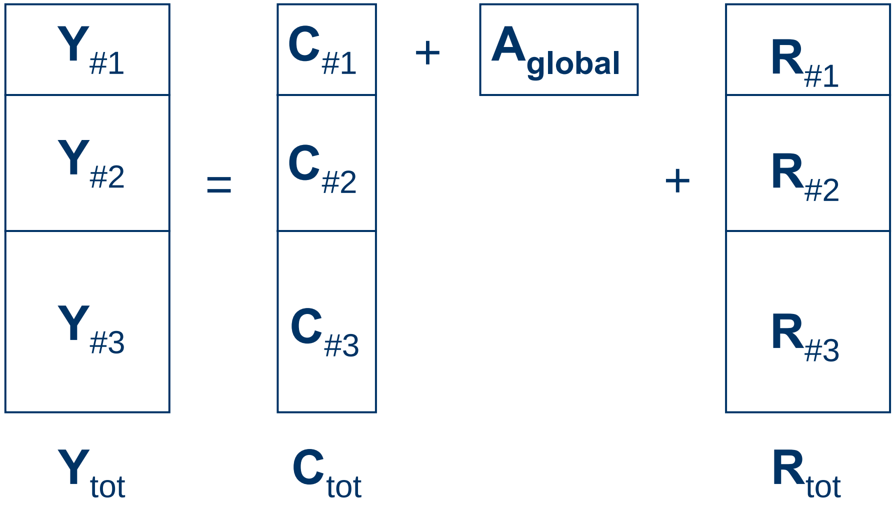
Simulated absorption spectra



Let’s make 3 Concentration dependent measurements

	A_0	B_0
# 1	*1*	*1*
# 2	*0.5*	*1*
# 3	*1*	*0.5*

Hypothesis of Global Spectra :
The 3 sets of experiment share the same pure spectra



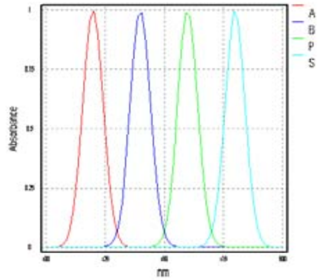
$Y_{\#1}$ $Y_{\#2}$ $Y_{\#3}$ = $C_{\#1}$ $C_{\#2}$ $C_{\#3}$ + A_{global} + $R_{\#1}$ $R_{\#2}$ $R_{\#3}$

Y_{tot} C_{tot} R_{tot}

Second Order Global Analysis

$$A+B \xrightarrow{k=0.5} P+S$$

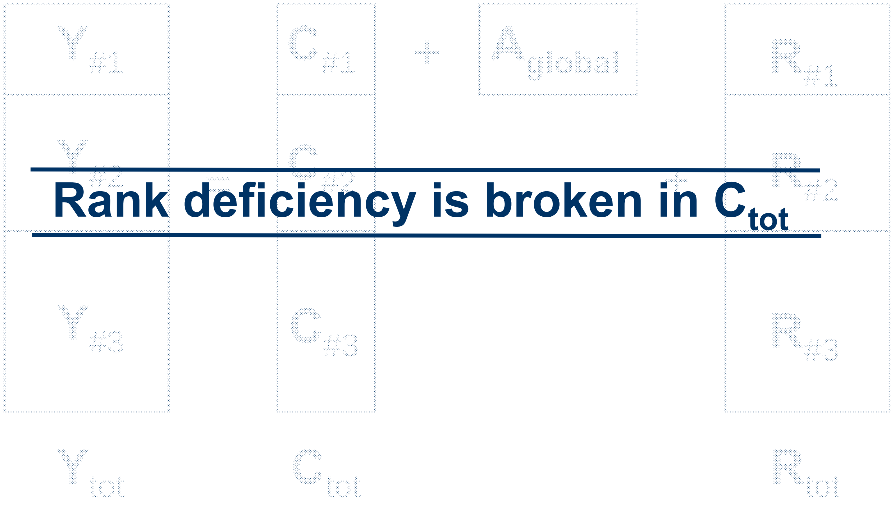
Simulated absorption spectra



Let’s make 3 Concentration dependent measurements

	A_0	B_0
# 1	*1*	*1*
# 2	*0.5*	*1*
# 3	*1*	*0.5*

Hypothesis of Global Spectra :
The 3 sets of experiment share the same pure spectra



$Y_{\#1}$ = $C_{\#1}$ + A_{global} + $R_{\#1}$

$Y_{\#2}$ = $C_{\#2}$ + $R_{\#2}$

$Y_{\#3}$ $C_{\#3}$ $R_{\#3}$

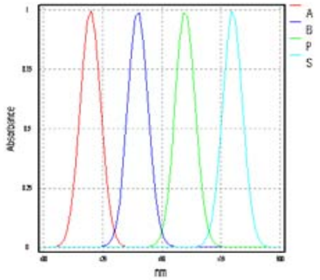
Y_{tot} C_{tot} R_{tot}

Rank deficiency is broken in C_{tot}

Second Order Global Analysis

$$A+B \xrightarrow{k=0.5} P+S$$

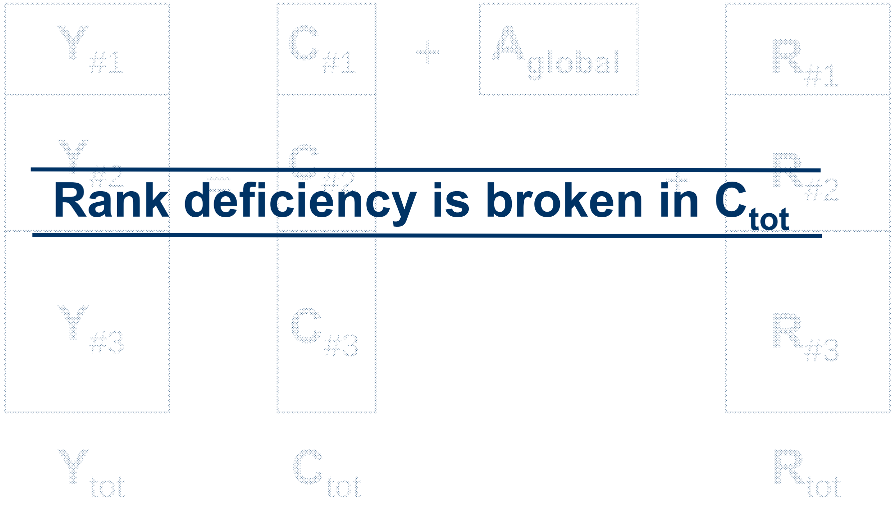
Simulated absorption spectra



Let’s make 3 Concentration dependent measurements

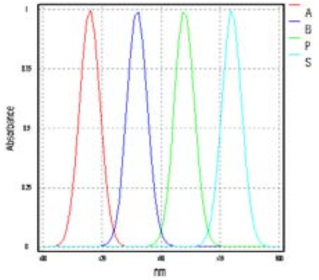
	A_0	B_0
# 1	*1*	*1*
# 2	*0.5*	*1*
# 3	*1*	*0.5*

Hypothesis of Global Spectra :
The 3 sets of experiment share the same pure spectra



Rank deficiency is broken in C_{tot}

Fitted absorption spectra :

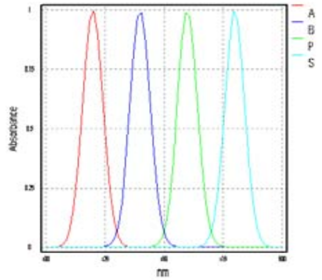


k_{fitted} = 0.5

Second Order Global Analysis

$$A + B \xrightarrow{k=0.5} P + S$$

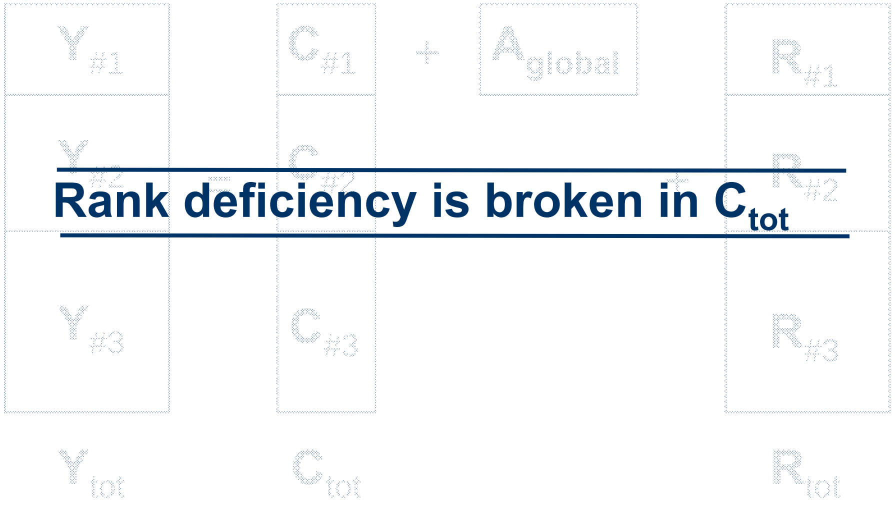
Simulated absorption spectra



Let’s make 3 Concentration dependent measurements

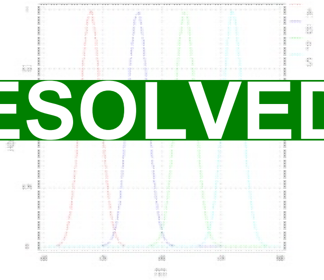
	A_0	B_0
# 1	*1*	*1*
# 2	*0.5*	*1*
# 3	*1*	*0.5*

Hypothesis of Global Spectra :
The 3 sets of experiment share the same pure spectra



Rank deficiency is broken in C_{tot}

Fitted absorption spectra :



RESOLVED

k_{fitted} = 0.5

End of the Tutorial

That is « already » the end
of this Tutorial

Thank you for your attention !

Case study overview

The aims of these case studies are :

- Identify the number of detectable species
- Know the phenomenon of parameter interchange
- Identify and break rank deficiency
- Use residuals for model validation

Using the following techniques :

- PCA
- EFA
- ALS
- Hard-modeling

On simulated data from different types :

- Spectroscopy
- Calorimetry

Under batch and semi-batch conditions

- $A \xrightarrow{k=0.5} P$
- $A \xrightarrow{k_1=0.5} B \xrightarrow{k_2=0.3} P$
- $A + B \xrightarrow{k=0.5} P$
- $A + B\left(dosed\right) \xrightarrow{k=0.5} P$
- $A + B\left(dosed\right) \xrightarrow{k_1=10} C$

  $B\left(dosed\right) + C \xrightarrow{k_2=5} P$

Case study 1

Mechanism : $A \xrightarrow{k=0.5} P$

Signal : Spectroscopy

Process : Batch

Conversion : 99%

- PCA/EFA : 2 species ⇒ Full rank
- ALS : Easily resolved
  with non-negativity constraint
- Hard-modeling : k does not depend on $c_0(A)$
  for a 1st order reaction

Case study 2

Mechanism	: $A \xrightarrow{k=0.5} P$
Signal	: Spectroscopy
Process	: Batch
Conversion	: 63%

- PCA/EFA : 2 species ⇒ Full rank
- ALS : Resolved under constraints
  with non-negativity constraint and known spectrum of *P*
- Hard-modeling : *k* does not depend on c_0(A)
  for a 1st order reaction

Case study 3

Mechanism	: $A \xrightarrow{k=0.5} P$
Signal	: Calorimetry
Process	: Batch
Conversion	: 99%

- PCA/EFA/ALS : NA
- Hard-modeling : *k* does not depend on $c_0(A)$ for a 1st order reaction

Case study 4

Mechanism	: $A + B \xrightarrow{k=0.5} P$
Signal	: Spectroscopy
Process	: Batch
Fitted mechanism	: $A \longrightarrow P$

- PCA/EFA : 2 species $\Rightarrow$ Full rank
- ALS : Apparently resolved under constraints

  with non-negativity constraint and known spectrum of *P*
- Hard-modeling : 1^{st} order mechanism is wrong !

Case study 5

Mechanism	:	$A \xrightarrow{k_1=0.5} B \xrightarrow{k_2=0.3} P$
Signal	:	Spectroscopy
Process	:	Batch
Conversion	:	99%

- PCA/EFA : 3 species ⇒ Full rank
- ALS : Resolved under constraints

  with non-negativity constraint and known spectrum of *B*
- Hard-modeling : *Parameter interchange*

  k_1 and k_2 swap depending on the initial guess

Case study 6

Mechanism : $A \xrightarrow{k_1=0.5} B \xrightarrow{k_2=0.3} P$

Signal : Spectroscopy

Process : Batch

Conversion : 77%

- PCA/EFA : 3 species ⇒ Full rank
- ALS : Hardly resolved

  with non-negativity constraint and known spectrum of *B* and *P*
- Hard-modeling : *Parameter interchange*

  k_1 and k_2 swap depending on the initial guess

Case study 7

Mechanism	:	$A \xrightarrow{k_1=0.5} B \xrightarrow{k_2=0.3} P$
Signal	:	Calorimetry
Process	:	Batch
Conversion	:	99%

- PCA/EFA/ALS : NA
- Hard-modeling : *1.* Less robust than multivariate fitting
  2. Parameter interchange

Case study 8

Mechanism : $A \xrightarrow{k_1=0.5} B \xrightarrow{k_2=0.3} P$

Signal : Spectroscopy

Process : Batch

Fitting : Univariate (λ = 400)

- PCA/EFA/ALS : NA
- Hard-modeling : *1.* Less robust than multivariate fitting
  2. Parameter interchange

Case study 9

Mechanism : $A \xrightarrow{k_1=0.5} B \xrightarrow{k_2=0.3} P$

Signal : Spectroscopy

Fitting : Univariate (λ = 400)

Fitted mechanism : $A \longrightarrow P$

- PCA/EFA/ALS : NA

- Hard-modeling : *1.* Less robust than multivariate fitting
  2. Parameter interchange
  3. Structured residuals

  ⇒ The model is slightly wrong but hard to validate at this single wavelength

Case study 10

Mechanism : $A + B \xrightarrow{k=0.5} P$

Signal : Spectroscopy

Process : Batch

Colored species : *A* and *P* (model reduction)

- PCA/EFA : 2 species ⇒ Rank deficiency
- ALS : NA
- Hard-modeling : Species *B* set as non-absorbing

⇒ *1. k* is correct

2. The fitted pure spectra are wrong
(linear combinations of the true pure spectra)

Case study 11

Mechanism	: $A + B \xrightarrow{k=0.5} P$
Signal	: Spectroscopy
Process	: Batch
Known spectrum	: Species *B*

- PCA/EFA : 2 species ⇒ Rank deficiency
- ALS : NA
- Hard-modeling : The pure spectrum of B is provided

⇒ *1. k* is correct

2. The fitted pure spectra are resolved

Case study 12

Mechanism : $A + B(dosed) \xrightarrow{k=0.5} P$

Signal : Spectroscopy

Process : Semibatch

Conversion : 68%

- PCA/EFA : 3 species ⇒ Full rank
- ALS : Resolved under strong constraints

  non-negativity constraint and known spectrum of *B* and *P* !
- Hard-modeling : The pure spectra are resolved

Case study 13

Mechanism	: $A + B\left(dosed\right) \xrightarrow{k=0.5} P$
Signal	: Calorimetry
Process	: Semibatch
Conversion	: 68%

- PCA/EFA/ALS : NA

- Hard-modeling : Fitting calorimetric data is more robust in semibatch than in batch conditions !

Case study 14

Mechanism :

$$A + B\left(dosed\right) \xrightarrow{k_1=10} C$$

$$B\left(dosed\right) + C \xrightarrow{k_2=5} P$$

Signal : Spectroscopy

Process : Semibatch

- PCA/EFA : 4 species ⇒ Full rank
- ALS : Resolved under very strong constraints

  non-negativity constraint and known spectrum of *B*, *C* and *P* !
- Hard-modeling : The pure spectra are resolved

Case study 15

Mechanism : $A + B(dosed) \xrightarrow{k_1=10} C$

$B(dosed) + C \xrightarrow{k_2=5} P$

Signal : Calorimetry

Process : Semibatch

- PCA/EFA/ALS : NA
- Hard-modeling : Fitting calorimetric data conditions is more robust in semibatch than in batch conditions !

Summary on the Case studies (1)

- PCA/EFA
  - These two techniques provide information on the number of observable species and therefore the maximum rank of **Y**

- ALS
  - Non-negativity constraint alone only resolves $A \rightarrow P$
  - Pure spectra of the products are required if they are not fully formed
  - Pure spectra of the intermediates are generally required for complete resolution
  - Without *a priori* knowledge on mechanisms and/or spectra, rank deficiency is undetectable

Summary on the Case studies (2)

- HARD-MODELING

BATCH CONDITIONS

- Multivariate fitting of spectrocopy data is more reliable than univariate fitting
- Fitting of calorimetric data is not very stable in batch conditions
- Rank deficiency due to the model can be easily broken by :
  - **Model reduction** (some species are set non-absorbing)
    Fitted pure spectra are wrong (linear combination of the true ones)
    but nonlinear parameters are correct !
  - **Known spectra provided**
    Fitted pure spectra and nonlinear parameters are in this case both correct

SEMIBATCH CONDITIONS

- Fitting of calorimetric data is more robust in semibatch conditions
- The dosing completely breaks simple rank deficient problems and partially highly complex mechanisms (eg. 2 intermediates, 2 products …)

FIRST ORDER MECHANISMS

- With 1st order mechanisms, rates are independent on initial concentrations and rate constants can swap without differences in fitting (*Parameter Interchange*)

Appendix 1 : List of the Matlab files

#	Simulated Mechanism	Fitted mechanism	Process	Model rank	Rank deficiency	Conversion	Fitted signal	Fitting	PCA / EFA	ALS	Hard-modeling	File
1	A -> P	same	Batch	2	No	99.3%	Spec	Multivariate	2	[> 0]	C_0(A) = 1 and 10	main_AtoP_X99
2	A -> P	same	Batch	2	No	63.2%	Spec	Multivariate	2	[> 0 , P]	NA	main_AtoP_X60
3	A -> P	same	Batch	2	No	99.3%	Cal	Univariate	NA	NA	NA	main_AtoP_cal
4	A + B -> P	A -> P	Batch	2	Yes	90.9%	Spec	Multivariate	2	NA	Structured residuals	main_ApBtoP_1st_order
5	A -> B -> P	same	Batch	3	No	99.9%	Spec	Multivariate	3	[> 0 , B]	Swap of k's	main_AtoBtoP_X99
6	A -> B -> P	same	Batch	3	No	77.7%	Spec	Multivariate	3	[> 0 , B , P]	NA	main_AtoBtoP_X75
7	A -> B -> P	same	Batch	3	No	99.9%	Cal	Univariate	NA	NA	Less robust than Case 5	main_AtoBtoP_cal
8	A -> B -> P	same	Batch	3	No	99.9%	Spec	Univariate	NA	NA	Less robust than Case 5	main_AtoBtoP_univar1
9	A -> B -> P	A -> B	Batch	3	No	99.9%	Spec	Univariate	NA	NA	Structure in the residuals	main_AtoBtoP_univar2
10	A + B -> P	same	Batch	2	Yes	90.9%	Spec	Multivariate	2	NA	B not absorbing	main_ApBtoP_batch
11	A + B -> P	same	Batch	2	Yes	90.9%	Spec	Multivariate	2	NA	Pure spectrum of B provided	main_ApBtoP_batch_Bknown
12	A + B(dosed) -> P	same	Semibatch	3	No	68.2%	Spec	Multivariate	3	[> 0 , B , P]	NA	main_ApBtoP_semi_Y
13	A + B(dosed) -> P	same	Semibatch	3	No	68.2%	Cal	Univariate	3	NA	NA	main_ApBtoP_semi_cal
14	A + B(dosed) -> C B(dosed) + C -> P	same	Semibatch	4	No	99.8%	Spec	Multivariate	4	[> 0 , B , C , P]	NA	main_ApBtoC_BpCtoP_semi_Y
15	A + B(dosed) -> C B(dosed) + C -> P	same	Semibatch	4	No	99.8%	Cal	Univariate	4	NA	NA	main_ApBtoC_BpCtoP_semi_cal