UCL Université catholique de Louvain

École polytechnique de Louvain (EPL)

epl

Symmetric Quasi-Newton Least Square

Developing a symmetrical version of the quasi-Newton Least Square algorithm

Dissertation presented by
Nicolas BOUTET

for obtaining the Master's degree in
Mathematical Engineering

Supervisor(s)
François GLINEUR, Rob HAELTERMAN

Reader(s)
Pierre-Antoine ABSIL, Nicolas ANDRIESSEN

Academic year 2016-2017

Contents

Abstract

The class of Least Change Secant Update Quasi-Newton (LCSU QN) methods is often used for root finding or for optimization problems.
The particularity of these methods is that the estimated Jacobian of the system is chosen satisfying a secant equation and such that it is the closest possible to the previous Jacobian.

In this study, we start from a particular LCSU QN method, Quasi-Newton Least Squares (QN-LS), which has been widely used to solve interaction problems recast as a root finding problem $\mathbf{f}(\mathbf{x}) = \mathbf{0}$. This method imposes that multiple secant equations are respected.
We turn our attention to the use of QN-LS on a new problem:$\min g(\mathbf{x})$. If $\mathbf{f}(\mathbf{x})$ is the gradient of $g(\mathbf{x})$ then the problem again becomes that of solving $\mathbf{f}(\mathbf{x}) = \mathbf{0}$. The Jacobian of $\mathbf{f}(\mathbf{x})$, i.e. Hessian of $g(\mathbf{x})$, is now symmetrical, something that is not taken in consideration in the original QN-LS method.

Given that QN-LS has similarities with Broyden's methods and given that Broyden's methods have been developed for symmetrical Jacobians, we consider the possibilities of developing a symmetrical variant of QN-LS.
We prove that this symmetrical variant only exists under limited conditions. We will therefore explore three different strategies in order to find an update formula for the estimated Jacobian, which fulfills as much as possible the symmetry and the multiple secants conditions.

Chapter 1

Introduction and problem statement

The evolution of numerical methods, the development of specific tools and the increasing computation power make it possible to solve heavier and more complex problems in multiple domains.

Being able to solve a given complex problem often leads to a new question: which value of the variables should we take in order to optimize some objective function?
Solving such an optimization problem corresponds to solving multiple instances of the original problem with different values of the variables. If the problem is heavy and complex, its optimization is therefore also far heavier and more complex.

Example 1.1. Let's take the example of the calculation of a blade (helicopter or drone blade, plane propeller, part of a turbine or a compressor...). We want to compute the value of a given property (strains, noise, turbine flow...).
The shape of the blade is determined by geometrical variables (for instance, a finite elements grid). Knowing this geometry, a specific solver can compute the air flow around the blade. Thanks to this air flow, the solver (probably another one), can compute the forces on the blade. Thanks to the forces, it is possible to compute the deformation of the blade (typically with a finite elements software). The deformation changes the geometry which impacts the air flow, and so on. The three calculation steps have to be executed multiple times in order to find the equilibrium, i.e. a combination of data such that the loop through the three solvers keeps the data unchanged.
Finally, at the equilibrium, we can compute the value of the objective function.
As we see, we have a high-dimensional complex problem.

For the optimization (e.g.: the minimization of the noise), this complex equilibrium calculation must be done at each iteration point of the optimization path. Intuitively, we can estimate that the choice of the best direction leads to n different equilibrium calculations, where n is the number of variables. As the variables are the coordinates of the points of the finite equilibrium grid, the problem leads to approximately n equilibrium evaluation for each step on the iteration path. It becomes a very huge calculation problem.

1.1 Optimization methods

There exists a lot of different methods to solve optimization problems.
The simplex algorithm (and its variants) is well known for linear programming (or some specific applications such as quadratic programming). But for complex problems, the algorithm is not usable.

For nonlinear programming, we use iterative methods. These methods move iteratively toward an optimal solution by evaluating or approximating the Hessians (or the gradients or only the function values), choosing a direction and a step length.
The method stops when one given convergence criterion is reached.

Basically, to minimize (or maximize) the function $g(\mathbf{x})$, the iterative methods try to reach the zeros of the gradient $\mathbf{f}(\mathbf{x}) = \nabla g(\mathbf{x})$. The optimization problem becomes thus a root finding problem on the gradient of the function.

1.2 Newton's method

The classical method for finding one root of a real function, $f(x) = 0$ for $f : \mathbb{R} \to \mathbb{R}$, is Newton's method (also called Newton-Raphson's method). The function is approximated by its tangent in a given point[1]:

$$f(x) \cong f(x_0) + f'(x_0)(x - x_0)$$

The algorithm then takes the intersection of the tangent with the x-axis as a new approximation of the root and the point for the next iteration of the algorithm.
This can be written as[1]:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

The method can also be generalized for n equations, i.e. to solve $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ for $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$. We then have:

$$\mathbf{x}_{i+1} = \mathbf{x}_i - J_f^{-1}(\mathbf{x}_i)\mathbf{f}(\mathbf{x}_i)$$

where $J_f(\mathbf{x})$ is the Jacobian:

$$J_f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}$$

This method is very efficient, but because of some practical problems, it is unusable in a lot of applications.
First, if the starting point is too far from the searched root, the algorithm can fall in a loop and never reach the solution, or even diverge!
A second even more important problem is that the Jacobian has to be calculated. Indeed, in a lot of applications, it is impossible to calculate this matrix. For instance, when we do not know $\mathbf{f}$ explicitly, the Jacobian has to be estimated. If the size of the system is too high, the calculation of the Jacobian (or the estimation of its value) takes too much time.

[1] Note that for this equation, we don't follow the convention defined in Chapter 2. Indeed, x_i is here the i-th element of a sequence of points x and not the i-th entry of a vector $\mathbf{x}$.

Next to that, the inversion of the Jacobian is also a costly operation. So Newton's method is not applicable in many situations.

1.3 Quasi-Newton method

To avoid calculating the Jacobian, Quasi-Newton methods replace it by a matrix B easier to compute. The relation is then basically:

$$\mathbf{x}_{i+1} = \mathbf{x}_i - B_i^{-1}\mathbf{f}(\mathbf{x}_i)$$

In this formulation, it is still needed to invert the matrix B which can still be a costly operation. Some methods avoid this inversion by directly considering the inverse of the Jacobian $H = B^{-1}$. This leads to the formula:

$$\mathbf{x}_{i+1} = \mathbf{x}_i - H_i\mathbf{f}(\mathbf{x}_i)$$

A given method often exists in both its direct (B_i) or "inverse" (H_i) version. B_i (resp. H_i) is an approximation that needs to be iteratively updated:

$$B_{i+1} = B_i + F(B_i, \mathbf{x}_i, \mathbf{x}_{i-1}, \ldots, \mathbf{x}_0)$$

There exists a lot a Quasi-Newton methods, the differences come mainly from the criteria used to calculate the update of B (resp. H). We will list some of these criteria in the next subsections. Note that next to these criteria, each Quasi-Newton method can also contain some parameters which leads to families of methods with variating parameters.

1.3.1 Small rank update

Having a small rank update of the Jacobian (or its inverse) should ensure a limited calculation time for the update.

1.3.2 Least change update

To ensure that the Jacobian does not change too much between two successive iterations, one of the criteria is to ensure that the approximate Jacobian at the new iteration is the "closest" to the approximate Jacobian of the previous found iteration.
Therefore, we need to define the distance between matrices. There are different possibilities to define the norm of the difference between two matrices. In this study, we will use the Frobenius norm ($\|B_i - B_{i-1}\|_{Fr}$) to calculate this distance.

1.3.3 Secant equation

In one dimension, a simple variation of the Newton's method is the secant method. The idea behind this method is to stop using the exact tangent of the linear approximation, and that point x_{i-1} should belong to the linear approximation of $f(x)$ in point x_i. In this method, the derivative of the function is approximated by[2]:

$$f'(x_i) \cong \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}$$

[2]Note that for this equation, we don't follow the convention defined in Chapter 2. Indeed, x_i is here the i-th element of a sequence of points x and not the i-th entry of a vector $\mathbf{x}$.

This is a Quasi-Newton method. In one dimension, the derivative is uniquely defined.
In n dimension, the secant equation is:

$$B_i(\mathbf{x}_i - \mathbf{x}_{i-1}) = \mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_{i-1}) \tag{1.1}$$

There are an infinite number of matrices B_i that respect this equation as B_i is a $n \times n$ matrix while equation (1.1) only defines n scalar equations.
Working on the inverse of B_i, H_i, we have:

$$H_i(\mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_{i-1})) = \mathbf{x}_i - \mathbf{x}_{i-1}$$

1.3.4 Multiple secant equations

The principle of the secant equation can be generalized to multiple secant equations. In n dimensions, we can force the linear approximation of the function in x_i to pass exactly through m preceding points (m smaller or equal to n). The equation is then:

$$B_i \left[\Delta\mathbf{x}_{i,1}|\Delta\mathbf{x}_{i,2}|\dots|\Delta\mathbf{x}_{i,m}\right] = \left[\Delta\mathbf{f}_{i,1}|\Delta\mathbf{f}_{i,2}|\dots|\Delta\mathbf{f}_{i,m}\right]$$

$$B_i \Delta X_i = \Delta F_i$$

where:

- $\Delta X_i = [\Delta\mathbf{x}_{i,1}|\Delta\mathbf{x}_{i,2}|\dots|\Delta\mathbf{x}_{i,m}]$ is a matrix $n \times m$ formed by the juxtaposition of m vectors.
- ΔF_i is built on the same way.
- $\Delta\mathbf{x}_{i,k} = \mathbf{x}_i - \mathbf{x}_{i-k}$
- $\Delta\mathbf{f}_{i,k} = \mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_{i-k})$
- $m \leq n$ the number of satisfied secant equations.

This is called the m secant equations condition.

1.3.5 Symmetry

In optimization, the problem is to find an optimum of a scalar function g. This is usually done by searching the root of the derivative (Jacobian) of the function $\mathbf{f} = \nabla g$.
What we until now called the Jacobian of the vector function $\mathbf{f}$ is then, in fact, the derivative of the Jacobian of the function g. This is the Hessian matrix:

$$B \cong \begin{pmatrix} \frac{\partial^2 g}{\partial x_1^2} & \cdots & \frac{\partial^2 g}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 g}{\partial x_1 \partial x_n} & \cdots & \frac{\partial^2 g}{\partial x_n^2} \end{pmatrix}$$

This matrix is symmetric. In the context of optimization, it could be interesting to have a matrix B (resp. H) being symmetric.

1.3.6 Semi-definite positiveness

To ensure to have an optimum and not a saddle point, the Hessian matrix should be positive (or negative definite). This is the case for a matrix M when for every $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x}^T M\mathbf{x} > 0$ (resp $\mathbf{x}^T M\mathbf{x} < 0$).
In optimization, it makes sense to ensure that the Hessian is positive (resp. negative) definite.

1.4 Problem-statement

1.4.1 Context

Within this study, we'll try to develop a method to solve high dimensional, costly computational optimization problems where the algebraic expression of the Hessian matrix isn't available. We explain the different terms:

- High dimensional means that there are a large number of variables.
- Costly computational is related to the fact that it takes a lot of time to compute one value of the objective function.
  - An obvious consequence is that the value of the gradient of the function is also costly to compute.
  - Note that if the algebraic expression of the gradient of the function isn't available (and it is obviously the case when there isn't any algebraic form for the function itself), it has to be approximated by means of multiple function calls.
- The algebraic expression of the Hessian matrix isn't available. So it has to be estimated in the scope of a Quasi-Newton Method.

Due to the cost of the gradient calls, our objective is to minimize the number of gradient calls during the optimization. Therefor we'll aim to maximize the amount of information that is used for the creation of the estimate of the Hessian matrix.

1.4.2 Scope of the study

The methods that we consider in this work are called "Least Change Secant Update" (LCSU). They are based on two properties:

1. m secant equations where $m = \min(i, \sigma)$, σ being a parameter $\in [1, n]$ that has to be chosen, i being the present iteration.
2. Least change update.

In [10], a method was developed for $\sigma = n$. The estimated Jacobian was created as a least squares approximation of the real Jacobian. This method has been called Quasi-Newton Least-Squares (QN-LS). Directly working on the inverse of the Jacobian instead of the Jacobian itself leads to QN-ILS (Inverse Least Squares)
QN-LS and QN-ILS, which we regroup under the abbreviation QN-(I)LS, have been widely used to solve interaction problems recast as a root finding problem $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ [10].

However, we turn our attention to a new problem: $\min|g(\mathbf{x})|$. If $\mathbf{f}$ is the gradient of g, then the problem becomes solving $\mathbf{f}(\mathbf{x}) = \mathbf{0}$. The gradient of $\mathbf{f}$ (i.e. Hessian of g) is now symmetrical. This was not taken in consideration in the original QN(I)-LS method.

The goal of this study is to develop a symmetric version of QN-(I)LS. This can be written as:

$$\text{“Find } B_i \in U = \left\{ A \in \mathbb{R}^{n \times n} \colon\ A\Delta X_i = \Delta F_i \text{ and } A = A^T \right\} \tag{1.2}$$

$$\text{that minimizes } \|B_i - B_{i-1}\|_{Fr} \text{ with } \sigma = n.\text{”}$$

The symmetric version of QN-ILS should also be developed, which leads to:

$$\text{“Find } H_i \in V = \left\{A \in \mathbb{R}^{n\times n}\text{: } A\Delta F_i = \Delta X_i \text{ and } A = A^T\right\} \tag{1.3}$$
$$\text{that minimizes } \|H_i - H_{i-1}\|_{Fr} \text{ with } \sigma = n\text{.”}$$

1.5 Outline of the study

In Chapter 2, we give the conventions, the important definitions, and some general theorems that will be used within this study.
Chapter 3 gives an overview of existing LCSU methods such a Symmetric Rank 1 (SR1), Broyden's Good and Bad Method, Powell-Symmetric-Broyden (PSB) and QN-LS.
In Chapter 4, we will develop the symmetric version of QN-LS and find out that it exists only under specific conditions. These conditions will be defined and analyzed in order to understand this phenomenon and how to take it into account in a generalization of the formula when the conditions aren't fulfilled.
In Chapter 5, we expose some of the existing strategies for this generalization, which we will try to tune up. We will then develop our own strategy for the generalization of the formula: a Symmetric LCSU with penalized older secants.
Finally, we will test, in Chapter 6, the methods exposed in Chapters 3 to 5 on multiple standard problems in order to compare their performances.

Chapter 2

Definitions and general theorems

2.1 Conventions

2.1.1 Notations

Unless otherwise formulated, we use the following conventions:

- Scalars are represented by small letters without subscript.
- Vectors are represented by bold small letters (e.g.: $\mathbf{v}$). By convention, a vector has one column and multiple rows.
- Matrices are represented by capital letters (e.g.: M).

We use the following notation for a subset or a transformation of a matrix:

- Entries of a vector are given by the same letter with one subscript (e.g.: v_i).
- Entries of a matrix are given by the corresponding small letter and two subscripts (e.g.: $m_{i,j}$).
- Alternatively, in particular if there are already subscripts for a matrix or a vector, we use the following notation: $m_{i,j} = [M]_{i,j}$
- Column i of matrix M: $\mathbf{m}_{*,i}$.
- The transposed matrix A is written A^T. In the same way, the transposed vector $\mathbf{u}$, is the row vector $\mathbf{u}^T$.
- The conjugate transposed matrix of A is noted $A^* = (\bar{A})^T = \bar{A^T}$, where $\bar{A}$ denote the matrix A with complex conjugated entries. The complex conjugate of $a + bi$, where a and b are reals, is $a - bi$.

Subscripts will also be used in case of iterations. For example, $\mathbf{v}_i$ and B_i are the i-th iteration of $\mathbf{v}$ and B. When the right subscript is already used, the iteration subscript will be put on the left: ${}_i\omega_j$ is the i-th iterate of ω_j, which itself is the j-th component of $\boldsymbol{\omega}$.

In the present study, unless stated otherwise, all functions, matrices, vectors, and scalars are real.

2.1.2 Products & norms

- For the scalar product (also called the inner product), we use: $\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{v}^* \mathbf{u}$. As we are working with real vectors, we have $\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{v}^T \mathbf{u}$.
- For the norm of a matrix, we use the Frobenius norm: $\|M\|_{Fr} = \sqrt{\sum_{i=1}^{l} \sum_{j=1}^{n} |m_{i,j}|^2}$.
- For the norm of a vector, we use the 2-norm: $\|\mathbf{v}\|_2$.

2.1.3 Vectorisation, juxtaposition

- The vectorization of a matrix converts the matrix into a column vector. For a $m \times n$ matrix A:
  $\mathtt{vec}(A) = [a_{1,1}, \ldots, a_{m,1}, a_{1,2}, \ldots, a_{m,2}, \ldots, a_{1,n}, \ldots a_{m,n}]^T \in \mathbb{R}^{n \times m}$.
- We define the juxtaposition of multiples vectors $\mathbf{x}_i$ as the matrix containing in its column the different vectors $\mathbf{x}_i$. It is written:

$$X = [\mathbf{x}_1 | \mathbf{x}_2 | \ldots | \mathbf{x}_n]$$

2.2 Definitions

2.2.1 Span and rank

- The span C_S of a set of vectors S is the set of all finite linear combinations of the vectors.
- The rank of a matrix is the dimension of the span of the set of its columns.
- If the rank of a matrix $M \in \mathbb{R}^{n \times n}$ is equal to n, the matrix is "full-rank".

2.2.2 Special matrices

2.2.2.1 Normal matrix

A normal matrix A is a complex square matrix that commutes with its conjugate transposed:

$$A^* A = A A^*$$

One can prove that a matrix A is normal if and only if there exists a diagonal matrix Λ and a unitary matrix U such that $A = U \Lambda U^*$ [22]. The diagonal entries of Λ are the eigenvalues of A, and the columns of U are the eigenvectors of A. The matching eigenvalues in Λ come in the same order as the eigenvectors in the columns of U.

2.2.2.2 Skew-symmetric matrix

A skew-symmetric matrix is a square matrix whose transpose is its negation.

$$A = -A^T$$

2.2.2.3 Row swapping matrix

A row swapping is an operation on a matrix A that switches all matrix elements on row i with their counterparts on row j.
The operation can be done by pre-multiplying the matrix A by a elementary swapping matrix. This elementary matrix P is obtained by swapping row i and row j of the identity matrix.

Swapping multiple rows is done by multiplying by multiple elementary swapping matrices:

$$P_k P_{k-1} \dots P_2 P_1 A$$

2.2.2.4 Projection

A projection is a linear transformation P from a vector space to itself such that $P^2 = P$. The matrix P is called a projection matrix.

2.2.3 Moore-Penrose pseudoinverse

The pseudoinverse of matrix $A \in \mathbb{C}^{m \times n}$ is a matrix $A^+ \in \mathbb{C}^{n \times m}$ such that:

- $AA^+A = A$
- $A^+AA^+ = A^+$
- $(AA^+)^* = AA^+$
- $(A^+A)^* = A^+A$

A^+ exists for any matrix A. In particular, when A has linearly independent columns, A^+ can be computed as:

$$A^+ = (A^*A)^{-1}A^*$$

This particular pseudoinverse constitutes a "left inverse", since, in this case, $A^+A = I$.
Of course as we work with $A \in \mathbb{R}^{m \times n}$, we have $A^+ = (A^TA)^{-1}A^T$.

2.2.4 Multidimensional derivatives

2.2.4.1 Gradient

The gradient is a multi-variable generalization of the derivative.
Let $g(\mathbf{x})$ be a function of several variables. The gradient is defined as:

$$\nabla g = (\frac{\partial g}{\partial x_1}, \frac{\partial g}{\partial x_2}, \dots \frac{\partial g}{\partial x_n})^T$$

2.2.4.2 Jacobian

The Jacobian is the matrix of all first-order partial derivatives of a vector-valued function.
Let $\mathbf{f}(\mathbf{x})$ be a vectorial function of several (possibly one single) variables. The Jacobian is defined

as:

$$J = \begin{bmatrix} \dfrac{\partial \mathbf{f}}{\partial x_1} & \cdots & \dfrac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_m}{\partial x_1} & \cdots & \dfrac{\partial f_m}{\partial x_n} \end{bmatrix}$$

2.2.4.3 Hessian

The Hessian matrix or Hessian is a square matrix of second-order partial derivatives of a scalar-valued function. It is the Jacobian matrix of the gradient of a multi-variable function.
Let $g(\mathbf{x})$ be a function of several variables. The Hessian is defined as:

$$H = \begin{bmatrix} \dfrac{\partial^2 g}{\partial x_1^2} & \dfrac{\partial^2 g}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 g}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 g}{\partial x_2 \partial x_1} & \dfrac{\partial^2 g}{\partial x_2^2} & \cdots & \dfrac{\partial^2 g}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 g}{\partial x_n \partial x_1} & \dfrac{\partial^2 g}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 g}{\partial x_n^2} \end{bmatrix}$$

This matrix is symmetric when the second partial derivatives are continuous.

2.3 Theorems and Lemmas

2.3.1 Rank on matrix

Lemma 2.1. Let $A \in \mathbb{R}^{n \times m}$, $\text{rank}(A) = 1$ if and only if there exists $\mathbf{u} \in \mathbb{R}^n$ and $\mathbf{v} \in \mathbb{R}^m$ such that $A = \mathbf{u}\mathbf{v}^T$

Proof
If we suppose $A = \mathbf{u}\mathbf{v}^T$. If $\mathbf{w} \in \mathbb{R}^m$, then $A\mathbf{w} = \mathbf{u}\mathbf{v}^T\mathbf{w} = (\mathbf{v}^T\mathbf{w})\mathbf{u}$. Thus, A maps every vector in $\mathbb{R}^m$ to a multiple of $\mathbf{v}$, so $\text{rank}(A) = 1$.

On the other hand, if $\text{rank}(A) = 1$, then for all $\mathbf{w} \in \mathbb{R}^m$, $A\mathbf{w} = k\mathbf{u}$ for some fixed $\mathbf{u} \in \mathbb{R}^n$. In particular, this is true for the basis vectors of $\mathbb{R}^m$, so every column of A is a multiple of $\mathbf{u}$. That is, $A = [v_1\mathbf{u}|v_2\mathbf{u}|\ldots|v_m\mathbf{u}] = \mathbf{u}\,[v_1|v_2|\ldots|v_m] = \mathbf{u}\mathbf{v}^T$.

QED

Lemma 2.2. Let A and $B \in \mathbb{R}^{n \times n}$, $\text{rank}(A + B) \leq$rank(A)+rank(B)

Proof Let C_A (resp. C_B, C_{A+B}) be the span of columns of A (resp. B, $A + B$).

$$\text{rank}(A + B) = \dim(C_{A+B}) \leq \dim(C_A) + \dim(C_B) = \text{rank}(A) + \text{rank}(B)$$

QED

Lemma 2.3. The rank of a sum of k matrices $A_i \in \mathbb{R}^{n\times n}$, such that $\text{rank}(A_i) = 1$ is at most k.

Proof By direct recursive application of Lemma 2.2 on rank-1 matrices.

QED

Lemma 2.4. The rank of a skew-symmetric real matrix is even.

Proof Let $A = -A^T$ a real skew-matrix.

A is a normal matrix. Indeed, as A is a real matrix, $A^* = A^T$. We have then:

$$A^*A = A^TA = (-A)(-A^T) = AA^T = AA^*$$

Then A can be diagonalized in a matrix Λ whose entries are the eigenvalues of A.
The rank of A is equal to the rank of Λ which is the number of non-zero eigenvalues of A.

The eigenvalues of A are found by solving $A\mathbf{u} = \lambda\mathbf{u}$. Taking the scalar product with $\mathbf{u}$ gives:

$$\lambda\langle\mathbf{u},\mathbf{u}\rangle = \langle\lambda\mathbf{u},\mathbf{u}\rangle = \langle A\mathbf{u},\mathbf{u}\rangle = \langle\mathbf{u},-A\mathbf{u}\rangle = \langle\mathbf{u},-\lambda\mathbf{u}\rangle = -\bar{\lambda}\langle\mathbf{u},\mathbf{u}\rangle$$

with $\bar{\lambda}$ is the complex conjugate of λ. So the eigenvalues are purely imaginary.
As the complex eigenvalues of a real matrix always comes in conjugate pairs, the number of eigenvalues different from zero is even and so is the rank of the matrix.

QED

2.3.2 Row swapping matrices

Lemma 2.5. Let P be a row swapping matrix, $P^{-1} = P^T$.

Proof Let P be an elementary swapping matrix. We then have $PP^T = I$, i.e. P is orthogonal. This is easy to check because when we multiply P with P^T the only times the one in the row matches with the one in a column is when the row and column number match, so $P^T = P^{-1}$.

If P swaps multiple rows, we can decompose $P = P_kP_{k-1}\dots P_2P_1$ and:

$$\begin{aligned} PP^T &= P_kP_{k-1}\dots P_2P_1(P_kP_{k-1}\dots P_2P_1)^T \\ &= P_kP_{k-1}\dots P_2P_1P_1^TP_2^T\dots P_{k-1}^TP_k^T \\ &= I \end{aligned}$$

QED

2.3.3 Equivalence of sets of differences of vectors

Lemma 2.6. Let $\mathbf{x}_i$ and $\mathbf{f}_i \in \mathbb{R}^n$ for $i = 0, 1, \ldots, m \leq n$.
We define:

- $\Delta\mathbf{x}_{i,k} = \mathbf{x}_i - \mathbf{x}_{i-k}$ and $\Delta\mathbf{f}_{i,k} = \mathbf{f}_i - \mathbf{f}_{i-k}$ for $k = 1, \ldots, i$.
- $\Delta X = [\Delta\mathbf{x}_{m,1}|\Delta\mathbf{x}_{m,2}|\ldots|\Delta\mathbf{x}_{m,m}]$, a matrix $n \times m$ formed by the stacking of m vectors. ΔF is built on the same way
- $\mathbf{s}_i = \mathbf{x}_i - \mathbf{x}_{i-1}$ and $\mathbf{y}_i = \mathbf{f}_i - \mathbf{f}_{i-1}$
- $S = [\mathbf{s}_1|\mathbf{s}_2|\ldots|\mathbf{s}_m]$ and Y is built on the same way

Then

$$A\Delta X = \Delta F \iff AS = Y$$

Proof We define the matrix T:

$$T = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 & -1 & 1 \\ 0 & 0 & 0 & \ldots & -1 & 1 & 0 \\ \vdots & \vdots & \vdots & \iddots & \iddots & \vdots & \vdots \\ \vdots & \vdots & \iddots & \iddots & \vdots & \vdots & \vdots \\ 0 & -1 & 1 & \ldots & 0 & 0 & 0 \\ -1 & 1 & 0 & \ldots & 0 & 0 & 0 \\ 1 & 0 & 0 & \ldots & 0 & 0 & 0 \end{bmatrix}$$

We have then $\Delta XT = S$ and $\Delta FT = Y$. By right multiplying the equation by T we have:

$$A\Delta XT = \Delta FT \Leftrightarrow AS = Y$$

QED

2.3.4 Inverse of matrix

2.3.4.1 Sherman-Morrison formula

The formula of Sherman-Morrison [21] can be used to express the inverse of a rank-one correction of a matrix for which we already know the inverse. The formula is:

Lemma 2.7.

$$(A + \mathbf{u}\mathbf{v}^T)^{-1} = A^{-1} - \frac{A^{-1}\mathbf{u}\mathbf{v}^T A^{-1}}{1 + \mathbf{v}^T A^{-1}\mathbf{u}}$$

Where A is a $n \times n$ regular matrix.

The proof of the formula is given in [21] or [24].

We note that it is possible that $(A + \mathbf{u}\mathbf{v}^T)^{-1}$ exists, but that the formula cannot be applied. It is the case if A is singular, but $(A + \mathbf{u}\mathbf{v}^T)$ is regular.

2.3.4.2 Woodbury matrix identity

The Woodbury matrix identity [26] is an extension of the Sherman-Morrison formula.

Lemma 2.8. For A a $n \times n$ regular matrix, U a $n \times k$ matrix, C a $k \times k$ regular matrix, and V a $k \times n$ matrix, we have:

$$(A+UCV)^{-1} = A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1}VA^{-1} \tag{2.1}$$

The proof of the identity can be found on [25].

This formula makes it possible to inverse a matrix corrected by multiple rank-one updates. Therefore, we must take:

- $C = I$.
- U is built as the stacking of column vector $\mathbf{u}$.
- V is built as the vertical stacking of line vector $\mathbf{v}^T$

For a sum of 2 rank 1 updates, this expression can be developed as:

$$\begin{aligned}(A + \mathbf{u}_1\mathbf{v}_1^T + \mathbf{u}_2\mathbf{v}_2^T)^{-1} = A^{-1} &- A^{-1}\frac{(\mathbf{v}_2^TA^{-1}\mathbf{u}_2)\mathbf{u}_1\mathbf{v}_1^T}{\Delta}A^{-1} - A^{-1}\frac{(\mathbf{v}_2^TA^{-1}\mathbf{u}_1)\mathbf{u}_2\mathbf{v}_1^T}{\Delta}A^{-1} \\ &- A^{-1}\frac{(\mathbf{v}_1^TA^{-1}\mathbf{u}_2)\mathbf{u}_1\mathbf{v}_2^T}{\Delta}A^{-1} - A^{-1}\frac{(\mathbf{v}_1^TA^{-1}\mathbf{u}_1)\mathbf{u}_2\mathbf{v}_2^T}{\Delta}A^{-1} \\ &+ A^{-1}\frac{\mathbf{u}_1\mathbf{v}_1^T}{\Delta}A^{-1} + A^{-1}\frac{\mathbf{u}_2\mathbf{v}_2^T}{\Delta}A^{-1}\end{aligned}$$

where $\Delta = 1 + \mathbf{v}_1^TA^{-1}\mathbf{u}_1 + \mathbf{v}_2^TA^{-1}\mathbf{u}_2 + \mathbf{v}_1^TA^{-1}\mathbf{u}_1\mathbf{v}_2^TA^{-1}\mathbf{u}_2 - \mathbf{v}_1^TA^{-1}\mathbf{u}_2\mathbf{v}_2^TA^{-1}\mathbf{u}_1$.

This form involves a lot of terms so, it is better to use the standard form (2.1) or to apply the Sherman-Morrison formula multiple times.
Note that for Woodbury, we still need to inverse a $k \times k$ matrix.

2.3.5 Identities involving pseudoinverse

Let $X \in \mathbb{R}^{n\times m}$, $Y \in \mathbb{R}^{n\times m}$, and $\mathbf{x} = X_{*,m}$ the last column of X, we give here some interesting results.

Lemma 2.9. Let $X \in \mathbb{R}^{n\times m}$, and $\mathbf{x} = X_{*,m}$ the last column of X, then:

$$XX^+\mathbf{x} = \mathbf{x}$$

$$\mathbf{x}^T(X^+)^TX^T = \mathbf{x}^T$$

Proof
$XX^+\mathbf{x}$ is the last column of XX^+X, but $XX^+X = X$. So $XX^+\mathbf{x}$ is the last column of X which is $\mathbf{x}$.
The second form is simply the transposed expression.

QED

Lemma 2.10. Let $X \in \mathbb{R}^{n\times m}$, and $\mathbf{x} = X_{*,m}$ the last column of X, then:

$$\mathbf{x}^T X X^+ = \mathbf{x}^T$$

$$(X^+)^T X^T \mathbf{x} = \mathbf{x}$$

Proof
$\mathbf{x}^T X X^+$ is the last row of $X^T X X^+$. But $X^T X X^+ = X^T X (X^T X)^{-1} X^T = X^T$. So $\mathbf{x}^T X X^+$ is the last row of X^T which is $\mathbf{x}^T$.
The second form is simply the transposed expression.

QED

Lemma 2.11. Let $X \in \mathbb{R}^{n\times m}$, $Y \in \mathbb{R}^{n\times m}$, $\mathbf{x} = X_{*,m}$ the last column of X, and $\mathbf{y} = Y_{*,m}$ the last column of Y, then:

$$Y X^+ \mathbf{x} = \mathbf{y}$$

$$\mathbf{x}^T (X^+)^T Y^T = \mathbf{y}^T$$

Proof
$Y X^+ \mathbf{x}$ is the last column of $Y X^+ X$. But $Y X^+ X = Y$. So $Y X^+ \mathbf{x}$ is the last column of Y which is $\mathbf{y}$.
The second form is simply the transposed expression.

QED

2.4 Standard formulation of problem statement

2.4.1 Standard parameters and variables

From now on, we will use the following definitions:

- n: the dimension of the problem to solve, the dimension of the space we work in.
- m: the number of secant equations that we will fulfill.
- $\mathbf{x}_i \in \mathbb{R}^n$: an iteration point in the Quasi-Newton process. The starting point is $\mathbf{x}_0$.
- $g(\mathbf{x}_i)$: The objective function to optimize evaluated in $\mathbf{x}_i$.
- $\mathbf{f}(\mathbf{x}_i) = \nabla g(\mathbf{x}_i)$: the gradient of the objective function evaluated in $\mathbf{x}_i$.
- $B_i \approx B(\mathbf{x}_i) = \nabla^2 g(\mathbf{x}_i) \in \mathbb{R}^{n\times n}$: the approximated Hessian of the objective function in $\mathbf{x}_i$. If no specific subscript, it is the last calculated Hessian.
- $S \in \mathbb{R}^{n\times m}$: the stacking of m column vectors $\mathbf{s}_j = \mathbf{x}_j - \mathbf{x}_{j-1}$ for $j = 1, \ldots, m$.
- $Y \in \mathbb{R}^{n\times m}$: the stacking of m column vectors $\mathbf{y}_j = \mathbf{f}(\mathbf{x}_j) - \mathbf{f}(\mathbf{x}_{j-1})$ $j = 1, \ldots, m$.
- $\mathbf{s} = \mathbf{s}_{*,m}$: the last column of S.
- $\mathbf{y} = \mathbf{y}_{*,m}$: the last column of Y.
- $W = Y - BS$
- $\mathbf{w} = \mathbf{y} - B\mathbf{s}$

- $H = B^{-1}$: Approximation of the inverse of the Hessian.
- $V = HY - S$
- $\mathbf{v} = H\mathbf{y} - \mathbf{s}$

For matrices and vectors of the list above, an additional subscript can be added to indicate at which iteration of the QN process we are (e.g.: S_i or $\mathbf{s}_i$). If omitted, the matrix or vector is the one corresponding to the present iteration of the QN process.

2.4.2 General hypothesis

S is a full rank matrix. If S is not full-rank, it is possible to filter S by deleting some columns such that it becomes full ranked. The corresponding columns of Y are then deleted, too.

Note that when working directly on the inverse Hessian (H_i), Y has to be full-rank.

2.4.3 Quasi-Newton steps

The Quasi-Newton optimization process starts with $\mathbf{x}_0$ as starting point and B_0 as an estimation of the Hessian matrix at point $\mathbf{x}_0$.
A first point $\mathbf{x}_1$ is chosen, basically by applying:

$$\mathbf{x}_1 = (1 - \omega)\mathbf{x}_0 - \omega B_0^{-1}\mathbf{f}(\mathbf{x}_0)$$

Or

$$\mathbf{x}_1 = (1 - \omega)\mathbf{x}_0 - \omega H_0\mathbf{f}(\mathbf{x}_0)$$

The use of a small $\omega > 0$ ensures $\mathbf{x}_1$ not to be too far away from $\mathbf{x}_0$.

The next points are calculated thanks to the optimization step, for $i = 1, 2, \ldots$:

$$\mathbf{x}_{i+1} = \mathbf{x}_i - B_i^{-1}\mathbf{f}(\mathbf{x}_i)$$

Or

$$\mathbf{x}_{i+1} = \mathbf{x}_i - H_i\mathbf{f}(\mathbf{x}_i)$$

We are looking for methods to calculate B_i (resp. H_i).

2.4.4 Problem statement

The problem expressed in (1.2) is then:

Given $B_i \in \mathbb{R}^{n\times n}$, S_i, and $Y_i \in \mathbb{R}^{n\times m}$ where $m = \min(i, n)$, we search

$$\arg\min_{B_{i+1}} \quad \tfrac{1}{2}\left\|B_i - B_{i+1}\right\|_{Fr}^2 \tag{2.2}$$

$$\text{such that} \quad B_{i+1}S_i - Y_i = 0 \tag{2.3}$$

$$B_{i+1} - B_{i+1}^T = 0 \tag{2.4}$$

Working on the inverse of the Hessian, as stated in (1.3), we have a second problem, which we call the dual form:

Given $H_i \in \mathbb{R}^{n\times n}$, S_i and $Y_i \in \mathbb{R}^{n\times m}$ where $m = \min(i, n)$, we search

$$\begin{aligned} \arg\min_{H_{i+1}} \quad & \tfrac{1}{2} \|H_i - H_{i+1}\|_{Fr}^2 && (2.5)\\ \text{such that} \quad & H_{i+1}Y_i - S_i = 0 && (2.6)\\ & H_{i+1} - H_{i+1}^T = 0 && (2.7) \end{aligned}$$

Equations (2.2) and (2.5) represent the least change property.
Equations (2.3) and (2.6) are the mutliple secant equations to fulfill, the use of S and Y instead of ΔX and ΔF is possible thanks to Lemma 2.6.
Equations (2.4) and (2.7) are the symmetrical.

Chapter 3

Existing methods

In this chapter, we give a small overview of existing methods. We start with rank-one updates and then move to more complicated models deriving from them.
For each existing method, we will check the desired properties (LCSU, multiple secant equations and symmetric). However, none of the formula combine all of the properties.

3.1 Quasi-Newton Algorithms

The algorithm consists in two main phases: the calculation of the new estimate of the Jacobian and the calculation of the new point. Different methods for updating the Jacobian B or its inverse H are exposed later in this chapter.

The calculation of a new point is typically done thanks to formulas:

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \rho_i B_i^{-1} \mathbf{f}(\mathbf{x}_i) \tag{3.1}$$

resp.

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \rho_i H_i \mathbf{f}(\mathbf{x}_i) \tag{3.2}$$

Within the framework of this study, we will only work with $\rho_i = 1$. Indeed, the selection of the most efficient value of ρ_i can be handled as a separate problem. Our goal is to compare methods to estimate B_i. We will avoid testing multiple sets of parameters.

To initialize the method, we first have to choose a starting point, $\mathbf{x}_0$, and a first estimation of the Jacobian, B_0.
However, it is often not sufficient. In most of the methods that we will list, the update is a function of $\mathbf{s}_i = \mathbf{x}_i - \mathbf{x}_{i-1}$ and $\mathbf{y}_i = \mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_{i-1})$. So, we also need a second point $\mathbf{x}_1$.
Choosing the second point can also be seen as a separate problem. Here again, we will not discuss the choice of $\mathbf{x}_1$ to avoid testing multiple parameters.

3.2 Rank-one updates

The majority of rank-one Quasi-Newton methods follow the standard form:

$$B_{i+1} = B_i + \frac{(\mathbf{y}_i - B_i \mathbf{s}_i)\mathbf{c}_i^T}{\mathbf{c}_i^T \mathbf{s}_i} = B_i + \frac{\mathbf{w}_i \mathbf{c}_i^T}{\mathbf{c}_i^T \mathbf{s}_i} \tag{3.3}$$

resp.

$$H_{i+1} = H_i + \frac{(\mathbf{s}_i - H_i\mathbf{y}_i)\mathbf{d}_i^T}{\mathbf{d}_i^T\mathbf{y}_i} = H_i + \frac{\mathbf{v}_i\mathbf{d}_i^T}{\mathbf{d}_i^T\mathbf{y}_i} \tag{3.4}$$

The choice of one of the two formulations and the particular value of vector $\mathbf{c}$ or $\mathbf{d}$ defines the method.

3.2.1 Symmetric rank-one update (SR1)

This method uses the unique update that is (1) rank-one, (2) symmetric, and (3) that respects the last secant equation.

Using (3.3), we already have a rank-one method. Defining $\mathbf{c} = \mathbf{w} = \mathbf{y}_i - B_i\mathbf{s}_i$, we force the symmetry of B_i. So, if this is enough to ensure the last secant equation, we have the formulation of SR1.

$$\begin{aligned} B_{i+1}\mathbf{s}_i &= B_i\mathbf{s}_i + \frac{(\mathbf{y}_i - B_i\mathbf{s}_i)\mathbf{w}_i^T}{\mathbf{w}_i^T\mathbf{s}_i}\mathbf{s}_i \\ &= B_i\mathbf{s}_i + (\mathbf{y}_i - B_i\mathbf{s}_i) \\ &= \mathbf{y}_i \end{aligned}$$

which proves that the update for SR1 is given by:

Formula 3.1 (SR1 - Direct Form)**.** Let $B_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let B_{i+1} such that:

- $\text{rank}(B_{i+1} - B_i) = 1$
- $B_{i+1} = B_{i+1}^T$
- $B_{i+1}\mathbf{s}_i = \mathbf{y}_i$

Then, B_{i+1} is given by:

$$B_{i+1} = B_i + \frac{(\mathbf{y}_i - B_i\mathbf{s}_i)(\mathbf{y}_i - B_i\mathbf{s}_i)^T}{(\mathbf{y}_i - B_i\mathbf{s}_i)^T\mathbf{s}_i} = B_i + \frac{\mathbf{w}_i\mathbf{w}_i^T}{\mathbf{w}_i^T\mathbf{s}_i}$$

Applying Lemma 2.7 and using the fact that H_i is already symmetric, we can express the inverse form of the formula.
We can also express the dual formula of SR1, the problem stated in (1.3).
We observe that both forms are the same:

Formula 3.2 (SR1 - Inverse Form)**.** Let $H_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let H_{i+1} such that:

- $\text{rank}(H_{i+1} - H_i) = 1$
- $H_{i+1} = H_{i+1}^T$
- $H_{i+1}\mathbf{y}_i = \mathbf{s}_i$

Then, H_{i+1} is given by:

$$H_{i+1} = H_i + \frac{(H_i\mathbf{y}_i - \mathbf{s}_i)(H_i\mathbf{y}_i - \mathbf{s}_i)^T}{(H_i\mathbf{y}_i - \mathbf{s}_i)^T\mathbf{y}_i} = H_i + \frac{\mathbf{v}_i\mathbf{v}_i^T}{\mathbf{v}_i^T\mathbf{y}_i}$$

This formula is a Secant Update, it is symmetric but it isn't, strictly speaking a Least Change update as the least change condition has not be imposed. However, the fact that the update formula is unique leads, de facto, to the fulfilling of the Least Change property.
Fially, it is, of course, a rank one update.

3.2.2 Broyden's first or "good" method (BG)

This method uses an update that is (1) rank-one, (2) that respects the last secant equation, and (3) such that the new Hessian is the closest to the previous one in the norm of Frobenius.

The formula has been proposed by Broyden in 1965 [6].

Formula 3.3 (BG - Direct Form)**.** Let $B_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let B_{i+1} such that:

- $\text{rank}(B_{i+1} - B_i) = 1$
- $B_{i+1}\mathbf{s}_i = \mathbf{y}_i$
- $\|B_{i+1} - B_i\|_{Fr}$ is minimal

Then, B_{i+1} is given by:

$$B_{i+1} = B_i + \frac{(\mathbf{y}_i - B_i\mathbf{s}_i)\mathbf{s}_i^T}{\mathbf{s}_i^T\mathbf{s}_i} = B_i + \frac{\mathbf{w}_i\mathbf{s}_i^T}{\mathbf{s}_i^T\mathbf{s}_i}$$

We see that the Formula 3.3 is a rank-one update.

The formula should therefor respect the properties (2) and (3). We give here our version of the proof:

Proof For more readability and to avoid to handle too many subscripts, we lightly change the notation in this development. We note:

- $B \equiv B_i$
- $A \equiv B_{i+1}$

We start with the following optimization problem:

$$\begin{aligned} \arg\min_A \quad & \tfrac{1}{2}\|A - B\|_{Fr}^2 \\ \text{such that} \quad & A\mathbf{s} - \mathbf{y} = 0 \end{aligned}$$

We take the Lagrangian of the system:

$$\mathcal{L}(A, \boldsymbol{\lambda}) = \frac{1}{2}\|A - B\|_{Fr}^2 + \sum_i \lambda_i(\sum_k A_{i,k}s_k - y_i)$$

We now take the partial derivative in function of $A_{i,j}$. We first note that:

$$\frac{\partial}{\partial A_{i,j}} \sum_i \lambda_i(\sum_k A_{i,k}s_k - y_i) = \lambda_i s_j$$

We find:

$$\frac{\partial \mathcal{L}}{\partial A_{i,j}} = A_{i,j} - B_{i,j} + \lambda_i s_j = 0$$

The system can thus be written as:

$$\begin{cases} A - B + \boldsymbol{\lambda}\mathbf{s}^T & = 0 \quad (3.5) \\ A\mathbf{s} - \mathbf{y} & = 0 \quad (3.6) \end{cases}$$

Putting (3.5) into (3.6), we find:

$$\begin{aligned} A\mathbf{s} - \mathbf{y} &= 0 \\ B\mathbf{s} - \boldsymbol{\lambda}\mathbf{s}^T\mathbf{s} - \mathbf{y} &= 0 \\ \boldsymbol{\lambda}\mathbf{s}^T\mathbf{s} &= B\mathbf{s} - \mathbf{y} \\ \boldsymbol{\lambda} &= \frac{B\mathbf{s} - \mathbf{y}}{\mathbf{s}^T\mathbf{s}} \end{aligned}$$

Putting this back into (3.5), we have:

$$A = B + \frac{(\mathbf{y} - B\mathbf{s})\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}}$$

which is the same form as Formula 3.3.

QED

This formula is a Least Change Secant Update, but it is not symmetric. Obviously, it is a rank one update.

Broyden also proposed to avoid inverting B by using the formula of Sherman-Morrison (Lemma 2.7). This leads to the inverse form:

Formula 3.4 (BG - Inverse Form)**.** Let $H_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let H_{i+1} such that:

- $\text{rank}(H_{i+1}^{-1} - H_i^{-1}) = 1$
- $H_{i+1}\mathbf{y}_i = \mathbf{s}_i$
- $\left\|H_{i+1}^{-1} - H_i^{-1}\right\|_{Fr}$ is minimal

Then, H_{i+1} is given by:

$$H_{i+1} = H_i + \frac{(\mathbf{s}_i - H_i\mathbf{y}_i)\mathbf{s}_i^T H_i}{\mathbf{s}_i^T H_i \mathbf{y}_i} = H_i + \frac{\mathbf{v}_i\mathbf{s}_i^T H_i}{\mathbf{s}_i^T H_i \mathbf{y}_i}$$

3.2.3 Broyden's second or "bad" method (BB)

The second Broyden's method is similar to the first one. Conditions (1) and (2) are kept. For the third condition, we take (3) the inverse of the Jacobian which is the closest to the previous one in the norm of Frobenius. It is the dual formula of Broyden's "good" method.
The formula is built in the same way as for Broyden's first method. This gives:

Formula 3.5 (BB - Inverse Form)**.** Let $H_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let H_{i+1} such that:

- $\text{rank}(H_{i+1} - H_i) = 1$
- $H_{i+1}\mathbf{y}_i = \mathbf{s}_i$
- $\|H_{i+1} - H_i\|_{Fr}$ is minimal

Then, H_{i+1} is given by:

$$H_{i+1} = H_i + \frac{(\mathbf{s}_i - H_i\mathbf{y}_i)\mathbf{y}_i^T}{\mathbf{y}_i^T\mathbf{y}_i} = H_i + \frac{\mathbf{v}_i\mathbf{y}_i^T}{\mathbf{y}_i^T\mathbf{y}_i}$$

The name "good" and "bad" initially comes from Broyden, because the first one seemed to give better results [5]. However, depending on the problem, the "bad" method can give better results than the "good" one. Both are used in more recent applications.
Note also that the inverse form of Broyden's first method isn't equivalent to the direct form of Broyden's second method as condition (3) is different.

3.3 Higher rank updates

3.3.1 Powell Symmetric Broyden (PSB)

In [17], Powell develops a symmetric version of Broyden's first method. This method thus combines the following properties:

(1) Symmetric;

(2) Respects the last secant equation;

(3) Such that the new Jacobian is the closest to the previous one in the norm of Frobenius.

The formula is:

Formula 3.6 (PSB)**.** Let $B_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let B_{i+1} such that:

- $B_{i+1} = B_{i+1}^T$
- $B_{i+1}\mathbf{s}_i = \mathbf{y}_i$
- $\|B_{i+1} - B_i\|_{Fr}$ is minimal

Then, B_{i+1} is given by:

$$B_{i+1} = B_i + \frac{\mathbf{w}_i\mathbf{s}_i^T}{\mathbf{s}_i^T\mathbf{s}_i} + \frac{\mathbf{s}_i\mathbf{w}_i^T}{\mathbf{s}_i^T\mathbf{s}_i} - \frac{\mathbf{w}_i^T\mathbf{s}_i}{(\mathbf{s}_i^T\mathbf{s}_i)^2}\mathbf{s}_i\mathbf{s}_i^T$$

In his article [17], Powell writes about "straightforward algebra" that leads to his formula. As we will use a similar development multiple times, we give here his proof and give more details about the development of this formula.

Proof The conditions (2) and (3) are fulfilled by the Broyden's good method (Formula 3.3). The condition (1) can be created by simply taking:

$$\bar{B} = \frac{1}{2}(B + B^T) \tag{3.7}$$

Powell proposed to build a sequence of matrices by alternating the two formulas: Formula 3.3 and formula (3.7) [17]. We start from a given matrix, apply the first formula, then apply the second formula to the result, then the first formula again and so on.
This recursive application of the two formulas is not detailed by Powell who simply calls it "straightforward algebra". We give here the full development.

We note ${}_jB$ (resp. ${}_j\bar{B}$) the j-th element of the sequence of B (resp. $\bar{B}$). We define ${}_j\mathbf{w} = \mathbf{y} - {}_jB\mathbf{s}$ and ${}_j\bar{\mathbf{w}} = \mathbf{y} - {}_j\bar{B}\mathbf{s}$. We start with ${}_0\bar{B}$ symmetric and we develop:

$$
\begin{aligned}
{}_1B &= {}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} \\
{}_1\bar{B} &= \frac{1}{2}({}_1B + {}_1B^T) \\
&= \frac{1}{2}({}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} + ({}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}})^T) \\
&= \frac{1}{2}({}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} + {}_0\bar{B}^T + \frac{\mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}}) \\
&= {}_0\bar{B} + \frac{1}{2}\frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T + \mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}} \\
{}_2B &= {}_1\bar{B} + \frac{{}_1\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} \\
&= {}_1\bar{B} + \frac{(\mathbf{y} - {}_1\bar{B}\mathbf{s})\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} \\
&= {}_1\bar{B}(I - \frac{\mathbf{s}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}}) + \frac{\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} \\
&= {}_0\bar{B} + \frac{1}{2}\frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T + \mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}} - ({}_0\bar{B} + \frac{1}{2}\frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T + \mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}})(\frac{\mathbf{s}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}}) + \frac{\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} \\
&= {}_0\bar{B} + \frac{1}{2}\frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T + \mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}} - (\frac{{}_0\bar{B}\mathbf{s}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} + \frac{1}{2}\frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T\mathbf{s}\mathbf{s}^T}{(\mathbf{s}^T\mathbf{s})^2} + \frac{1}{2}\frac{\mathbf{s}_0\bar{\mathbf{w}}^T\mathbf{s}\mathbf{s}^T}{(\mathbf{s}^T\mathbf{s})^2}) + \frac{\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} \\
&= {}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{2\mathbf{s}^T\mathbf{s}} + \frac{\mathbf{s}_0\bar{\mathbf{w}}^T}{2\mathbf{s}^T\mathbf{s}} - \frac{{}_0\bar{B}\mathbf{s}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} - \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T\mathbf{s}\mathbf{s}^T}{2(\mathbf{s}^T\mathbf{s})^2} - \frac{\mathbf{s}_0\bar{\mathbf{w}}^T\mathbf{s}\mathbf{s}^T}{2(\mathbf{s}^T\mathbf{s})^2} + \frac{\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} \\
&= {}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{2\mathbf{s}^T\mathbf{s}} + \frac{\mathbf{s}_0\bar{\mathbf{w}}^T}{2\mathbf{s}^T\mathbf{s}} - \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{2\mathbf{s}^T\mathbf{s}} - \frac{\mathbf{s}_0\bar{\mathbf{w}}^T\mathbf{s}\mathbf{s}^T}{2(\mathbf{s}^T\mathbf{s})^2} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} \\
&= {}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} + \frac{\mathbf{s}_0\bar{\mathbf{w}}^T}{2\mathbf{s}^T\mathbf{s}} - \frac{{}_0\bar{\mathbf{w}}^T\mathbf{s}}{2(\mathbf{s}^T\mathbf{s})^2}\mathbf{s}\mathbf{s}^T \\
{}_2\bar{B} &= {}_0\bar{B} + \frac{3}{4}\frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T + \mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}} - \frac{1}{2}\frac{{}_0\bar{\mathbf{w}}^T\mathbf{s}}{(\mathbf{s}^T\mathbf{s})^2}\mathbf{s}\mathbf{s}^T \\
{}_3B &= {}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} + \frac{3}{4}\frac{\mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}} - \frac{3}{4}\frac{{}_0\bar{\mathbf{w}}^T\mathbf{s}}{(\mathbf{s}^T\mathbf{s})^2}\mathbf{s}\mathbf{s}^T
\end{aligned}
$$

Going on in the same way, leads to:

$$
\begin{aligned}
{}_n\bar{B} &= {}_0\bar{B} + \frac{(2^n - 1)}{2^n}\frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T + \mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}} - \frac{2^{n-1} - 1}{2^{n-1}}\frac{{}_0\bar{\mathbf{w}}^T\mathbf{s}}{(\mathbf{s}^T\mathbf{s})^2}\mathbf{s}\mathbf{s}^T \\
{}_{n+1}B &= {}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} + \frac{2^n - 1}{2^n}\frac{\mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}} - \frac{2^n - 1}{2^n}\frac{{}_0\bar{\mathbf{w}}^T\mathbf{s}}{(\mathbf{s}^T\mathbf{s})^2}\mathbf{s}\mathbf{s}^T
\end{aligned}
$$

Taking the limit to infinity, we see that the sequence converges to:

$${}_\infty\bar{B} = {}_\infty B = {}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} + \frac{\mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}^T\mathbf{s}} - \frac{{}_0\bar{\mathbf{w}}^T\mathbf{s}}{(\mathbf{s}^T\mathbf{s})^2}\mathbf{s}\mathbf{s}^T$$

So, if B_i is symmetric and we want a symmetric LCSU, the update formula must be:

$$B_{i+1} = B_i + \frac{\mathbf{w}_i\mathbf{s}_i^T}{\mathbf{s}_i^T\mathbf{s}_i} + \frac{\mathbf{s}_i\mathbf{w}_i^T}{\mathbf{s}_i^T\mathbf{s}_i} - \frac{\mathbf{w}_i^T\mathbf{s}_i}{(\mathbf{s}_i^T\mathbf{s}_i)^2}\mathbf{s}_i\mathbf{s}_i^T$$

QED

Note that, in fact, Powell used in the proof above the principle of alternating projections. This principle will be explained more in details and used in Chapter 4.

3.3.2 Inverse PSB

Applying the same development as in previous section but using $H_i = B_i^{-1}$ instead of B_i leads to the dual version of PSB.

To our knowledge, this formula has never been published. However, as the formula comes immediately from the development of the previous section, we give the formula here:

Formula 3.7 (IPSB)**.** Let $H_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let H_{i+1} such that:

- $H_{i+1} = H_{i+1}^T$
- $H_{i+1}\mathbf{y}_i = \mathbf{s}_i$
- $\|H_{i+1} - H_i\|_{Fr}$ is minimal

Then, H_{i+1} is given by:

$$H_{i+1} = H_i + \frac{\mathbf{v}_i\mathbf{y}_i^T}{\mathbf{y}_i^T\mathbf{y}_i} + \frac{\mathbf{y}_i\mathbf{v}_i^T}{\mathbf{y}_i^T\mathbf{y}_i} - \frac{\mathbf{v}_i^T\mathbf{y}_i}{(\mathbf{y}_i^T\mathbf{y}_i)^2}\mathbf{y}_i\mathbf{y}_i^T$$

3.3.3 Quasi-Newton-Least Square (QN-LS)

This method, developed in [10], fulfills 2 conditions: (1) it respects multiple secant equations and (2) the new Jacobian is the closest to the previous one in the norm of Frobenius.

The formula is based on B_i respecting the $i-1$ preceding secant equations. B_{i+1} must then additionally respect the i-th secant equation.

Formula 3.8 (QN-LS)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and S_i full-ranked. Let B_{i+1} such that:

- $B_{i+1}S_i = Y_i$
- $\|B_{i+1} - B_i\|_{Fr}$ is minimal

Then, B_{i+1} is given by:

$$B_{i+1} = B_i + \frac{(\mathbf{y}_i - B_i\mathbf{s}_i)((I - L_{i-1}L_{i-1}^T)\mathbf{s}_i)^T}{\mathbf{s}_i^T(I - L_{i-1}L_{i-1}^T)\mathbf{s}_i} = B_i + \frac{\mathbf{w}_i\mathbf{s}_i^T(I - L_{i-1}L_{i-1}^T)}{\mathbf{s}_i^T(I - L_{i-1}L_{i-1}^T)\mathbf{s}_i}$$

Where $L_{i-1} \in \mathbb{R}^{n\times m}$ contains in its columns a orthogonal basis of the range of ΔX_{i-1}.

The proof of the LCSU property is given in [10].

The multisecant proprety is easily verified.
Proof For $j = i$, we have:

$$B_{i+1}\mathbf{s}_i = B_i\mathbf{s}_i + \frac{\mathbf{w}_i\mathbf{s}_i^T(I - L_{i-1}L_{i-1}^T)\mathbf{s}_i}{\mathbf{s}_i^T(I - L_{i-1}L_{i-1}^T)\mathbf{s}_i} = B_i\mathbf{s}_i + \mathbf{w}_i = B_i\mathbf{s}_i + \mathbf{y}_i - B_i\mathbf{s}_i = \mathbf{y}_i$$

For $j = 1, \ldots, i-1$, we know that $B_i\mathbf{s}_j = \mathbf{y}_j$. We also note the following:

- As L_{i-1} is orthogonal, $L_{i-1}^{-1} = L_{i-1}^T$.
- $L_{i-1}L_{i-1}^T$ is a projection matrix. Indeed,

$$(L_{i-1}L_{i-1}^T)^2 = L_{i-1}L_{i-1}^{-1}L_{i-1}L_{i-1}^T = L_{i-1}L_{i-1}^T$$

- As L_{i-1} is orthogonal, $\mathbf{u}_j = L_{i-1}^{-1}\mathbf{v}_j$ is a change of basis. $\mathbf{v}_j$ are the coordinates in the canonical basis and $\mathbf{u}_j$ are the coordinates in new basis. On the same way $\mathbf{w}_j = L_{i-1}\mathbf{u}_j$ is the change of basis back to the original space.
- The size of the intermediate space, the one with coordinates $\mathbf{u}_j$, is $i - 1 < n$. The change of basis is also a projection in a new smaller space. In fact, $L_{i-1}L_{i-1}^{-1}$ is a mapping from a space of dimension n to a space of dimension $i - 1$.
- $\mathbf{s}_j$ belongs the image of L_{i-1} (it is a direct consequence of the way L_{i-1} has been constructed). So, L_{i-1}^{-1} maps $\mathbf{s}_j$ to itself in the new basis and $L_{i-1}L_{i-1}^{-1}\mathbf{s}_j = \mathbf{s}_j$.

Applying this to the formula gives:

$$B_{i+1}\mathbf{s}_j = B_i\mathbf{s}_j + \frac{\mathbf{w}_i\mathbf{s}_i^T(I - L_{i-1}L_{i-1}^T)\mathbf{s}_j}{\mathbf{s}_i^T(I - L_{i-1}L_{i-1}^T)\mathbf{s}_i} = B_i\mathbf{s}_j + \frac{\mathbf{w}_i\mathbf{s}_i^T(\mathbf{s}_j - \mathbf{s}_j)}{\mathbf{s}_i^T(I - L_{i-1}L_{i-1}^T)\mathbf{s}_i} = \mathbf{y}_j$$

QED

The name "Least Squares" comes from the fact that the formula can be built by trying to construct the approximate Hessian using a Least Squares approach based on known input-output pairs of the Jacobian.
The method is a LCSU, it respects multiple secant equations but it is not symmetric.

3.3.4 QN-ILS

This method is equivalent to QN-LS but, it is using $H = B^{-1}$ instead of B.

Formula 3.9 (QN-ILS)**.** Let $H_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and S_i full-ranked. Let H_{i+1} such that:

- $H_{i+1}Y_i = S_i$
- $\|H_{i+1} - H_i\|_{Fr}$ is minimal

Then, H_{i+1} is given by:

$$H_{i+1} = H_i + \frac{(\mathbf{s}_i - H_i\mathbf{y}_i)((I - K_{i-1}K_{i-1}^T)\mathbf{y}_i)^T}{\mathbf{y}_i^T(I - K_{i-1}K_{i-1}^T)\mathbf{y}_i} = H_i + \frac{\mathbf{v}_i\mathbf{y}_i^T(I - K_{i-1}K_{i-1}^T)}{\mathbf{y}_i^T(I - K_{i-1}K_{i-1}^T)\mathbf{y}_i}$$

Where K_{i-1} contains in its columns a orthogonal basis of the range of ΔF_{i-1}.

3.3.5 Other methods

In this section, we list some methods that are out of the scope of the study. Indeed, they all fulfill the definite positiveness condition for the Hessian. This is not what we impose and, therefore, these methods are outside the scope of our study.

3.3.5.1 Davidon–Fletcher–Powell (DFP)

This formula provides new estimate of the Hessian that is (1) symmetric, (2) positive definite, and (3) closest to the previous approximate value of the Hessian.
For B_i already positive definite, the update is given by:

Formula 3.10 (DFP - Direct Form)**.** Let $B_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let B_{i+1} such that:

- $B_{i+1} = B_{i+1}^T$
- B_{i+1} is positive definite
- $B_{i+1}\mathbf{s}_i = \mathbf{y}_i$
- $\|B_{i+1} - B_i\|_{Fr}$ is minimal

Then, B_{i+1} is given by:

$$B_{i+1} = (I - \frac{\mathbf{y}_i\mathbf{s}_i^T}{\mathbf{y}_i^T\mathbf{s}_i})B_i(I - \frac{\mathbf{s}_i\mathbf{y}_i^T}{\mathbf{y}_i^T\mathbf{s}_i}) + \frac{\mathbf{y}_i\mathbf{y}_i^T}{\mathbf{y}_i^T\mathbf{s}_i} = B_i - \frac{\mathbf{y}_i\mathbf{s}_i^T B_i}{\mathbf{y}_i^T\mathbf{s}_i} - \frac{B_i\mathbf{s}_i\mathbf{y}_i^T}{\mathbf{y}_i^T\mathbf{s}_i} + \frac{\mathbf{y}_i\mathbf{s}_i^T B_i\mathbf{s}_i\mathbf{y}_i^T}{(\mathbf{y}_i^T\mathbf{s}_i)^2} + \frac{\mathbf{y}_i\mathbf{y}_i^T}{\mathbf{y}_i^T\mathbf{s}_i}$$

Using the formula of Sherman-Morrison (Lemma 2.7), we can also express the inverse form:

Formula 3.11 (DFP - Inverse Form)**.** Let $H_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let H_{i+1} such that:

- $H_{i+1}^{-1} = (H_{i+1}^{-1})^T$
- H_{i+1}^{-1} is positive definite
- $H_{i+1}\mathbf{y}_i = \mathbf{s}_i$
- $\left\|H_{i+1}^{-1} - H_i^{-1}\right\|_{Fr}$ is minimal

Then, H_{i+1} is given by:

$$H_{i+1} = H_i - \frac{H_i\mathbf{y}_i\mathbf{y}_i^T H_i}{\mathbf{y}_i^T H_i\mathbf{y}_i} + \frac{\mathbf{s}_i\mathbf{s}_i^T}{\mathbf{y}_i^T\mathbf{s}_i}$$

3.3.5.2 Broyden–Fletcher–Goldfarb–Shanno (BFGS)

This formula provides new estimate of the inverse of the Hessian that is (1) symmetric, (2) positive definite, and (3) closest to the previous approximate value of the inverse of the Hessian. This is the dual form of DFP.
For H_i already positive definite, the update is given by:

Formula 3.12 (BFGS - Direct Form)**.** Let $H_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$. Let H_{i+1} such that:

- $H_{i+1} = H_{i+1}^T$
- H_{i+1} is positive definite
- $H_{i+1}\mathbf{y}_i = \mathbf{s}_i$
- $\|H_{i+1} - H_i\|_{Fr}$ is minimal

Then, H_{i+1} is given by:

$$H_{i+1} = (I - \frac{\mathbf{s}_i\mathbf{y}_i^T}{\mathbf{s}_i^T\mathbf{y}_i})H_i(I - \frac{\mathbf{y}_i\mathbf{s}_i^T}{\mathbf{s}_i^T\mathbf{y}_i}) + \frac{\mathbf{s}_i\mathbf{s}_i^T}{\mathbf{s}_i^T\mathbf{y}_i} = H_i - \frac{\mathbf{s}_i\mathbf{y}_i^T H_i}{\mathbf{s}_i^T\mathbf{y}_i} - \frac{H_i\mathbf{y}_i\mathbf{s}_i^T}{\mathbf{s}_i^T\mathbf{y}_i} + \frac{\mathbf{s}_i\mathbf{y}_i^T H_i\mathbf{y}_i\mathbf{s}_i^T}{(\mathbf{s}_i^T\mathbf{y}_i)^2} + \frac{\mathbf{s}_i\mathbf{s}_i^T}{\mathbf{s}_i^T\mathbf{y}_i}$$

This update formula is currently widely used in optimization.

3.4 Conclusion

In this chapter, we gave an overview of some existing methods which are related to our problem. We give a summary of these methods and their properties in Table 3.1.
However, none of the existing methods combine all the desired properties (LCSU, multiple secant equations and symmetric). So, in the next chapter, we will combine some of the existing methods in order to create a new one fulfilling all the conditions.

Method	Formula	Least Change		Secant Update		Other Secants		Symmetry		Other properties
		On B	On H	No	Last secant	No	Multiple secants	No Symmetry	Symmetric	
SR1	3.1				X	X			X	Rank 1
BG	3.3	X			X	X		X		
BB	3.5		X		X	X		X		
PSB	3.6	X			X	X			X	
IPSB	3.7		X		X	X			X	
QN-LS	3.8	X			X		X	X		
QN-ILS	3.9		X		X		X	X		
DFP	3.10	X			X	X			X	Positive Definite
BFGS	3.12		X		X	X			X	Positive Definite

Table 3.1: Summary table of the existing methods discussed in this chapter.

Chapter 4

Toward Symmetric Quasi-Newton Least Square

We will now try to develop an update formula that satisfies the following properties:

(1) Symmetric;

(2) satisfies multiple secant equations;

(3) The new estimate of the Hessian is the closest to the previous one.

As the dimension of the problem is n, the Hessian matrix contains n^2 entries. So, for the determination of the new Hessian matrix, we have n^2 scalar entries.
Each vectorial secant equation defines n different scalar equations. So m secant equations define $n \times m$ scalar equations. The symmetry condition adds $\frac{(n-1)n}{2}$ new equations. We have in total $n(m + \frac{n-1}{2})$ equations.
When $n^2 > n(m + \frac{n-1}{2})$ or more easily $n + 1 > 2m$, the system is under-determined. The under-determinacy can be reduced by imposing the least change condition. So, the problem seems to be easy to solve.
However, we will see that this solution doesn't exist unless $Y^T S = S^T Y$, i.e. $Y^T S$ is symmetric.

In this chapter, we will use a simple approach to find the wanted formula. This method called the alternating projection will be explained in Section 4.1.
This method will be applied in Section 4.2 to find a symmetric version of QN-LS.
We will then prove (in Section 4.3) that this formula only exists under specific conditions. We will define the necessary and sufficient conditions for the existence of the formula, and analyze the reason why the formula exists only under specific conditions, which is against the conclusion of the easy reasoning above.
Finally, observing that the results in Section 4.2 don't fully include the Secant Update property, we will try to improve our results in Section 4.4. possib

4.1 Alternating projections

We will use the principle of alternating projections. This method is based on the successive projections P_i of a point x on different subspaces (K_i). The projection is the map that associates the point x to the point $P_i x \in K_i$, the closest to x.
In [7] and [4], it is proven that, if K_1 and K_2 are closed convex sets in a Hilbert space, the alternating projection converges to a fixed point that belongs to the intersection of K_1 and K_2.

In our case, working with matrices, we have to project on closed convex subsets in $\mathbb{R}^{n\times n}$. $\mathbb{R}^{n\times n}$ is a Euclidean space, which is a special case of a Hilbert space.

4.1.1 Closed convex sets

In this chapter, we will basically project on 3 sets:

- $K_S = \left\{A \in \mathbb{R}^{n\times n} : A = A^T\right\}$: The set of symmetric matrices.
- $K_{MS} = \{A \in \mathbb{R}^{n\times n} : AS_i = Y_i\}$: The set of matrices satisfying multiple given secant equations (we call them multisecant matrices).
- $K_{SU} = \{A \in \mathbb{R}^{n\times n} : A\mathbf{s}_i = \mathbf{y}_i\}$: The set of matrices satisfying one given secant equation (we call them SU matrices). Note that $K_{MS} \subset K_{SU}$.

We prove now that these sets are closed and convex.

Lemma 4.1. The set of symmetric matrices is a closed convex set.

Proof

- Closed: The set of symmetric matrices is closed under the operation of forming linear combinations, so it is a subspace [1]. Indeed, let A and B be two symmetrical matrices. We have, $\forall\lambda$:

  $$(\lambda A+(1-\lambda)B)^T = (\lambda A)^T+((1-\lambda)B)^T = \lambda A^T+(1-\lambda)B^T = \lambda A+(1-\lambda)B = (\lambda A+(1-\lambda)B)$$

  We have a finite dimensional subspace in a Hilbert space, so it is closed.
- Convex: We simply apply the equations used for the closed part $\forall\lambda \in [0, 1]$.

QED

Lemma 4.2. The set of multisecant matrices is a closed convex set.

Proof

- Closed: The system $AS = Y$ can be rewritten as $S_{\texttt{new}}\texttt{vec}(A) = F_{\texttt{new}}$ by simply writing the scalar equations of $AS = Y$ under each other. This system can also we written as the combination of inequalities: $S_{\texttt{new}}\texttt{vec}(A) \leq F_{\texttt{new}}$ and $S_{\texttt{new}}\texttt{vec}(A) \geq F_{\texttt{new}}$.
  The set $P = \left\{x \in \mathbb{R}^{n^2} : S_{\texttt{new}}x \leq F_{\texttt{new}} \texttt{ and } S_{\texttt{new}}x \geq F_{\texttt{new}}\right\}$ is a polyhedron in $\mathbb{R}^{n^2}$. But this polyhedron contains all its limit points, so it is closed.
- Convex: Let A and $B \in \mathbb{R}^{n\times n}$ such that $AS = Y$ and $BS = Y$ for S and $Y \in \mathbb{R}^{n\times m}$ given. We have, $\forall\lambda \in [0, 1]$:

  $$(\lambda A + (1 - \lambda)B)S = \lambda AS + (1 - \lambda)BS = \lambda Y + (1 - \lambda)Y = Y$$

QED

Lemma 4.3. The set of SU matrices is a closed convex set.

Proof This is an application of Lemma 4.2 where we only take one column of S and Y.

QED

4.1.2 Projections

The alternating projection creates a sequence of matrices, alternatively one matrix belonging to K_1 and then one belonging to K_2. We note ${}_jB$, the j-th projection of B on one of the loci K_i. We will sometimes add an extra subscript or another indication to indicate the locus onto we project (e.g. ${}_jB_{K_1}$ for the projection on K_1, ${}_jB_{MS}$ for the projection on multisecant matrices, ${}_j\bar{B}$ for the projection on symmetric matrices...).
In the same way, we define ${}_jW = Y - {}_jBS$.

The sequence of projection is thus:

$$ {}_0B \to {}_1B_{K_1} \to {}_1B_{K_2} \to {}_2B_{K_1} \to {}_2B_{K_2} \to {}_3B_{K_1} \to {}_3B_{K_2} \dots $$

We give in the following subsections three different projections that we will use in this chapter.

4.1.2.1 Projection on the set of symmetric matrices

In [14], it is shown that the minimal projection onto the space of symmetric matrices is given by $P(A) = \frac{A+A^T}{2}$, if the norm is symmetric (which is the case for the Frobenius norm). This is the projection on K_S:

$$ {}_j\bar{B} = \frac{1}{2}({}_jB + {}_jB^T) \tag{4.1} $$

We also define ${}_j\bar{W} = Y - {}_j\bar{B}S$.

4.1.2.2 Projection on the set of SU matrices

For the projection on K_{SU}, we need a projection on the set of matrices satisfying one secant equation and minimizing the Frobenius norm. This is by definition the formula of Broyden's "good" method (Formula 3.3).

4.1.2.3 Projection on the set of multisecant matrices

The last projection must ensure the multiple secant equations. We need to use the QN-LS update, which is the projection, closest to the previous estimate in the Frobenius norm, that satisfies multiple secant equations [10].
We cannot use Formula 3.8 because it projects onto the space of matrices satisfying n secant equations but only from the subspace of matrices satisfying $n-1$ secant equations. This will generally not be the case.
For this reason, we need to develop another update formula.

The formula for the projection on K_{MS} is:

Formula 4.1 (QN-LS - General Form)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and S_i full-rank. Let B_{i+1} such that:

- $B_{i+1}S_i = Y_i$
- $\|B_{i+1} - B_i\|_{Fr}$ is minimal

Then, B_{i+1} is given by:

$$B_{i+1} = B_i + (Y_i - B_iS_i)(S_i^TS_i)^{-1}S_i^T$$

Proof For more readability and to avoid to handle too many subscripts, we lightly change the notation in this development. We note:

- $A \equiv B_{i+1}$
- $B \equiv B_i$
- We omit the subscript i for S and Y.
- j, k the scalar coordinates within a vector or a matrix (j-th row, k-th column).

We start with the following optimization problem:

$$\begin{array}{ll} \arg\min\limits_{A} & \frac{1}{2}\|A-B\|_{Fr}^2 \\ \text{such that} & AS - Y \quad = 0 \end{array}$$

We take the Lagrangian of the system:

$$\mathcal{L}(A,\Lambda) = \frac{1}{2}\|A-B\|_{Fr}^2 + \sum_{j,k}\Lambda_{j,k}(\sum_l A_{j,l}S_{l,k} - Y_{j,k})$$

We now take the partial derivative in function of $A_{j,k}$. We first note that:

$$\begin{aligned} \frac{\partial}{\partial A_{j,k}}\sum_{j,k}\Lambda_{j,k}(\sum_l A_{j,l}S_{l,k} - Y_{j,k}) &= \frac{\partial}{\partial A_{j,k}}\sum_{j,k,l}\Lambda_{j,k}A_{j,l}S_{l,k} \\ &= \frac{\partial}{\partial A_{j,k}}\sum_{j,l,k}\Lambda_{j,l}A_{j,k}S_{k,l} \\ &= \sum_l \Lambda_{j,l}S_{k,l} \\ &= (\Lambda S^T)_{j,k} \end{aligned}$$

We find:

$$\frac{\partial\mathcal{L}}{\partial A_{j,k}} = A_{j,k} - B_{j,k} + (\Lambda S^T)_{j,k} = 0$$

The system can thus be written as:

$$\begin{cases} A - B + \Lambda S^T \quad = 0 & (4.2) \\ AS - Y \quad = 0 & (4.3) \end{cases}$$

Putting (4.2) into (4.3), we find:

$$\begin{aligned} AS - Y &= 0 \\ BS - \Lambda S^TS - Y &= 0 \\ \Lambda S^TS &= BS - Y \\ \Lambda &= (BS - Y)(S^TS)^{-1} \end{aligned}$$

Putting this back into (4.2), we have a new update formula:

$$A = B + (Y - BS)(S^T S)^{-1} S^T$$

$$B_{i+1} = B_i + (Y_i - B_i S_i)(S_i^T S_i)^{-1} S_i^T$$

Formulas 3.8 and 4.1 (QN-LS) are both from the same optimization problem. This problem is convex: the norm is strictly convex by property and there are only affine constraints. Therefor, the problem has only one solution and the two formulas are equivalent.

QED

The form of Formula 4.1 (QN-LS) is very similar to the Broyden's "good" method (Formula 3.3). We can see it as a generalized version of Broyden's method (we project onto K_{MS} instead of K_{SU}).
As PSB projects onto $K_{SU} \cap K_S$ and the symmetric QN-LS formula we are looking for, must project onto $K_{MS} \cap K_S$, it can be seen as a generalized version of PSB. From now on we will call it generalized PSB (gPSB).

4.2 Generalized PSB

4.2.1 PSB

We can now justify the method used by Powell for the development of PSB (Section 3.3.1).
He used a alternating projection with:

- Broyden's "good" method (Formula 3.3) for the projection on the set of SU matrices (K_{SU}).
- Equation (4.1) for the projection on the set of symmetric matrices (K_S).

4.2.2 Alternating projection to generalized PSB

We are now looking for the formula of generalized PSB. The problem can be expressed as:
Let $B_i \in \mathbb{R}^{n \times n}$, Y_i and $S_i \in \mathbb{R}^{n \times m}$, with $m \leq n$ and S_i full-rank. We search for B_{i+1} such that:

- $B_{i+1} S_i = Y_i$
- $B_{i+1} = B_{i+1}^T$

To find the formula of gPSB, we will now project alternatively on the locus of multisecant matrices (K_{MS}) and on the locus of symmetric matrices (K_S). We use thus:

- K_{MS}: The QN-LS formula, Formula 4.1 (QN-LS), for the projection on the set of multisecant matrices. We call the projection ${}_jB$.
- K_S: Equation (4.1) for the projection on the set of symmetric matrices. We call the projection ${}_j\bar{B}$.

We recall that $(S^T S)^{-1} S^T = S^+$ (See Section 2.2.3). We start with ${}_0B$ and we develop:

$$
\begin{aligned}
{}_1B &= {}_0B - {}_0BSS^+ + YS^+ \\
{}_1\bar{B} &= \frac{{}_1B}{2} + \frac{{}_1B^T}{2} \\
&= \frac{{}_0B^T}{2} + \frac{{}_0B}{2} - \frac{{}_0BSS^+}{2} - \frac{(S^+)^T S^T {}_0B^T}{2} + \frac{(S^+)^T Y^T}{2} + \frac{YS^+}{2}
\end{aligned}
$$

After those two first projections, we go on:

$$
\begin{aligned}
{}_2B &= {}_1B - {}_1BSS^+ + YS^+ \\
&= \frac{{}_0B^T}{2} - \frac{{}_0B^TSS^+}{2} + \frac{{}_0B}{2} - \frac{{}_0BSS^+}{2} - \frac{(S^+)^T S^T {}_0B^T}{2} \\
&\quad + \frac{(S^+)^T S^T {}_0B^TSS^+}{2} + \frac{(S^+)^T Y^T}{2} - \frac{(S^+)^T Y^T SS^+}{2} + YS^+
\end{aligned}
$$

$$
\begin{aligned}
{}_2\bar{B} &= \frac{1}{2}({}_2B + {}_2B^T) \\
&= \frac{{}_0B^T}{2} - \frac{1}{4}{}_0B^TSS^+ + \frac{{}_0B}{2} - \frac{{}_0BSS^+}{2} - \frac{(S^+)^T S^T {}_0B^T}{2} + \frac{1}{4}(S^+)^T S^T {}_0B^TSS^+ - \frac{1}{4}(S^+)^T S^T {}_0B \\
&\quad + \frac{1}{4}(S^+)^T S^T {}_0BSS^+ - \frac{1}{4}(S^+)^T S^T YS^+ + \frac{3}{4}(S^+)^T Y^T - \frac{1}{4}(S^+)^T Y^T SS^+ + \frac{3}{4}YS^+
\end{aligned}
$$

$$
\begin{aligned}
{}_3B &= \frac{{}_0B^T}{2} - \frac{{}_0B^TSS^+}{2} + \frac{{}_0B}{2} - \frac{{}_0BSS^+}{2} - \frac{(S^+)^T S^T {}_0B^T}{2} + \frac{(S^+)^T S^T {}_0B^TSS^+}{2} \\
&\quad - \frac{1}{4}(S^+)^T S^T {}_0B + \frac{1}{4}(S^+)^T S^T {}_0BSS^+ + \frac{3}{4}(S^+)^T Y^T - \frac{3}{4}(S^+)^T Y^T SS^+ + YS^+
\end{aligned}
$$

$$
\begin{aligned}
{}_3\bar{B} &= \frac{{}_0B^T}{2} - \frac{3}{8}{}_0B^TSS^+ + \frac{{}_0B}{2} - \frac{{}_0BSS^+}{2} - \frac{(S^+)^T S^T {}_0B^T}{2} + \frac{3}{8}(S^+)^T S^T {}_0B^TSS^+ - \frac{3}{8}(S^+)^T S^T {}_0B \\
&\quad + \frac{3}{8}(S^+)^T S^T {}_0BSS^+ - \frac{3}{8}(S^+)^T S^T YS^+ + \frac{7}{8}(S^+)^T Y^T - \frac{3}{8}(S^+)^T Y^T SS^+ + \frac{7}{8}YS^+
\end{aligned}
$$

$$
\begin{aligned}
{}_4B &= \frac{{}_0B^T}{2} - \frac{{}_0B^TSS^+}{2} + \frac{{}_0B}{2} - \frac{{}_0BSS^+}{2} - \frac{(S^+)^T S^T {}_0B^T}{2} + \frac{(S^+)^T S^T {}_0B^TSS^+}{2} \\
&\quad - \frac{3}{8}(S^+)^T S^T {}_0B + \frac{3}{8}(S^+)^T S^T {}_0BSS^+ + \frac{7}{8}(S^+)^T Y^T - \frac{7}{8}(S^+)^T Y^T SS^+ + YS^+
\end{aligned}
$$

Going on on the same way leads to:

$$
\begin{aligned}
{}_jB &= \frac{{}_0B^T}{2} - \frac{{}_0B^TSS^+}{2} + \frac{{}_0B}{2} - \frac{{}_0BSS^+}{2} - \frac{(S^+)^T S^T {}_0B^T}{2} + \frac{(S^+)^T S^T {}_0B^TSS^+}{2} \\
&\quad - \frac{2^{j-2}-1}{2^{j-1}}(S^+)^T S^T {}_0B + \frac{2^{j-2}-1}{2^{j-1}}(S^+)^T S^T {}_0BSS^+ \\
&\quad + \frac{2^{j-1}-1}{2^{j-1}}(S^+)^T Y^T - \frac{2^{j-1}-1}{2^{j-1}}(S^+)^T Y^T SS^+ + YS^+
\end{aligned}
$$

$$
\begin{aligned}
{}_j\bar{B} &= \frac{{}_0B^T}{2} - \frac{2^{j-1}-1}{2^j}{}_0B^TSS^+ + \frac{{}_0B}{2} - \frac{{}_0BSS^+}{2} - \frac{(S^+)^T S^T {}_0B^T}{2} \\
&\quad + \frac{2^{j-1}-1}{2^j}(S^+)^T S^T {}_0B^TSS^+ - \frac{2^{j-1}-1}{2^j}(S^+)^T S^T {}_0B + \frac{2^{j-1}-1}{2^j}(S^+)^T S^T {}_0BSS^+ \\
&\quad - \frac{2^{j-1}-1}{2^j}(S^+)^T S^T YS^+ + \frac{2^j-1}{2^j}(S^+)^T Y^T - \frac{2^{j-1}-1}{2^j}(S^+)^T Y^T SS^+ + \frac{2^j-1}{2^j}YS^+
\end{aligned}
$$

Taking the limit to infinity, we see that the sequences converge to 2 different formulas.
On one side, we have $\lim_{j\to\infty} {}_j\bar{B}$, the (1) symmetric formula (2) closest to the space of matrices satisfying multiple secant equations:

Formula 4.2 (gPSB Sym)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and S_i full-rank. Let K_{ScMS} be the set of matrices B such that:

- B is symmetric
- B is the closest to the set $K_{MS} = \{A \in \mathbb{R}^{n\times n} : AS_i = Y_i\}$

Then, $B_{i+1} \in K_{ScMS}$ such that $\|B_{i+1} - B_i\|_{Fr}$ is minimal is given by:

$$B_{i+1} = \frac{B_i^T}{2} - \frac{B_i^T S_i S_i^+}{2} + \frac{B_i}{2} - \frac{B_i S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T B_i^T}{2} + \frac{(S_i^+)^T S_i^T B_i^T S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T B_i}{2} + \frac{(S_i^+)^T S_i^T B_i S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T Y_i S_i^+}{2} + (S_i^+)^T Y_i^T - \frac{(S_i^+)^T Y_i^T S_i S_i^+}{2} + Y_i S_i^+$$

The second formula, $\lim_{j\to\infty} {}_jB$, gives the matrix (1) satisfying multiple secant equations and (2) being the closest to the symmetric matrix:

Formula 4.3 (gPSB MS)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and S_i full-rank. Let K_{MScS} be the set of matrices B such that:

- $BS_i = Y_i$
- B is the closest to the set $K_S = \left\{A \in \mathbb{R}^{n\times n} : A = A^T\right\}$

Then, $B_{i+1} \in K_{MScS}$ such that $\|B_{i+1} - B_i\|_{Fr}$ is minimal is given by:

$$B_{i+1} = \frac{B_i^T}{2} - \frac{B_i^T S_i S_i^+}{2} + \frac{B_i}{2} - \frac{B_i S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T B_i^T}{2} + \frac{(S_i^+)^T S_i^T B_i^T S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T B_i}{2} + \frac{(S_i^+)^T S_i^T B_i S_i S_i^+}{2} + (S_i^+)^T Y_i^T - (S_i^+)^T Y_i^T S_i S_i^+ + Y_i S_i^+$$

This two formulas are different. This means that the loci don't have any point in common (see illustration on Figure 4.1).

4.2.3 Dual forms

We can, of course, express the dual formulas. We apply the same development as in the previous section but we work with $H = B^{-1}$ instead of B and we invert S_i and Y_i. The "inversed" sets need to be defined: for instance, $K_{IMS} = \{A \in \mathbb{R}^{n\times n} : AY_i = S_i\}$. K_{IMS} (Inverse MultiSecant) is the equivalent of K_{MS} where S_i and Y_i have been inverted.

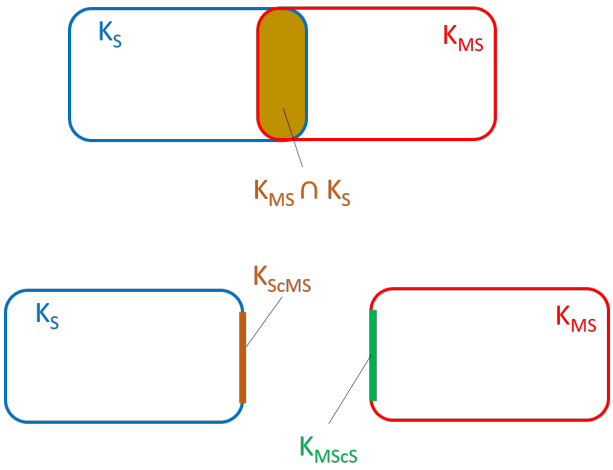

Figure 4.1:
Above: Intersection of loci K_S and K_{MS}.
Below: As the loci don't have any point in common, the alternating projection converge to two different sets K_{ScMS} and K_{MScS}. The two sets are projected onto each other.

Formula 4.4 (IgPSB Sym)**.** Let $H_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and Y_i full-rank. Let K_{IScMS} be the set of matrices H such that:

- H is symmetric
- H is the closest to the set $K_{IMS} = \{A \in \mathbb{R}^{n\times n} : AY_i = S_i\}$

Then, $H_{i+1} \in K_{IScMS}$ such that $\|H_{i+1} - H_i\|_{Fr}$ is minimal is given by:

$$\begin{aligned} H_{i+1} = &\frac{H_i^T}{2} - \frac{H_i^T Y_i Y_i^+}{2} + \frac{H_i}{2} - \frac{H_i Y_i Y_i^+}{2} - \frac{(Y_i^+)^T Y_i^T H_i^T}{2} \\ &+ \frac{(Y_i^+)^T Y_i^T H_i^T Y_i Y_i^+}{2} - \frac{(Y_i^+)^T Y_i^T H_i}{2} + \frac{(Y_i^+)^T Y_i^T H_i Y_i Y_i^+}{2} \\ &- \frac{(Y_i^+)^T Y_i^T S_i Y_i^+}{2} + (Y_i^+)^T S_i^T - \frac{(Y_i^+)^T S_i^T Y_i Y_i^+}{2} + S_i Y_i^+ \end{aligned}$$

Formula 4.5 (IgPSB MS)**.** Let $H_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and Y_i full-rank. Let K_{IMScS} be the set of matrices H such that:

- $HY_i = S_i$
- H is the closest to the set $K_S = \left\{A \in \mathbb{R}^{n\times n} : A = A^T\right\}$

Then, $H_{i+1} \in K_{IMScS}$ such that $\|H_{i+1} - H_i\|_{Fr}$ is minimal is given by:

$$\begin{aligned} H_{i+1} = &\frac{H_i^T}{2} - \frac{H_i^T Y_i Y_i^+}{2} + \frac{H_i}{2} - \frac{H_i Y_i Y_i^+}{2} - \frac{(Y_i^+)^T Y_i^T H_i^T}{2} + \frac{(Y_i^+)^T Y_i^T H_i^T Y_i Y_i^+}{2} \\ &- \frac{(Y_i^+)^T Y_i^T H_i}{2} + \frac{(Y_i^+)^T Y_i^T H_i Y_i Y_i^+}{2} + (Y_i^+)^T S_i^T - (Y_i^+)^T S_i^T Y_i Y_i^+ + S_i Y_i^+ \end{aligned}$$

4.3 Impossibility of problem-statement

In 1983, Schnabel gave a formula for the gPSB [20]:

$$B_{i+1} = B_i + Y_i S_i^+ - B_i S_i S_i^+ + (S_i^+)^T Y_i - (S_i^+)^T S_i^T B_i^T - (S_i^+)^T Y_i^T S_i S_i^+ + (S_i^+)^T S_i^T B_i^T S_i S_i^+ \quad (4.4)$$

This formula is equivalent to Formula 4.3 (gPSB MS) when B_i is symmetric. In the same publication, he proved that his formula only exists if $Y_i^T S_i$ is symmetric.

We give a more general theorem about the necessary and sufficient conditions of the existence of gPSB. We develop two different proofs. The first one comes directly from formulas 4.2 (gPSB Sym) and 4.3 (gPSB MS). The second one aims to explain the apparent paradox of the impossibility of solving a system that seems under-determined (having more variables than equations).

Theorem 4.1. Let Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and S_i full-rank. There exists $B_{i+1} \in \mathbb{R}^{n\times n}$ such that:

- $B_{i+1} S_i = Y_i$
- B_{i+1} is symmetric

if and only if $Y_i^T S_i$ is symmetric.

Proof 1 - Based on formulas 4.2 (gPSB Sym) and 4.3 (gPSB MS)
$\Rightarrow$ If the solution B_{i+1} exists, there is an intersection between the locus of matrices satisfying multiple secant equations and the locus of symmetric matrices. The alternating projections should converge to an unique formula. We equalize the 2 formulas and simplify terms appearing on both sides, we have then:

$$-\frac{(S_i^+)^T S_i^T Y_i S_i^+}{2} - \frac{(S_i^+)^T Y_i^T S_i S_i^+}{2} = -(S_i^+)^T Y_i^T S_i S_i^+$$

This last equation must be for every S_i and Y_i. This leads to the conditions $S_i^T Y_i = Y_i^T S_i$

$\Leftarrow$ If $Y_i^T S_i$ is symmetric, the formulas 4.2 (gPSB Sym) and 4.3 (gPSB MS) are equal. Both forms are symmetric and satisfy multiple secant equations, so there is a solution to the problem.

QED

Proof 2 - Linear dependence
We are looking for $B \in \mathbb{R}^{n\times n} : BS = Y$ and $B = B^T$ with S and $Y \in \mathbb{R}^{n\times m}$.

Step 1: construction of S^{syst}
We notice that $B\mathbf{s}_{*,1} = \mathbf{y}_{*,1}$ can be expressed as $S^1\mathbf{b} = \mathbf{y}_{*,1}$ where

- $\mathbf{b} = \texttt{vec}(B)$, a column vector containing every elements of B.
- S^1 a block diagonal $n \times n^2$ matrix containing the line vector $\mathbf{s}_{*,1}^T$ in each block of the diagonal.
- $\mathbf{y}_{*,1}$ still being a column vector containing the first column of Y.

We create S^i and $\mathbf{y}_{*,i}$ on the same way for the following columns of S and Y.

We now express the symmetry condition on the same form: $\Sigma\mathbf{b} = \mathbf{0}$.

Σ is a $\frac{n(n-1)}{2} \times n^2$ matrix. For each couple $\{b_{ij}, b_{ji}\}$ $(1 \leq i \leq n-1,\ i+1 \leq j \leq n)$, we have a line containing 0 everywhere but at the place corresponding to b_{ij} and b_{ji} where the value is respectively 1 and -1.

Applying the 2 conditions $BS = Y$ and $B = B^T$ together, we have thus:

$$[(S^1)^T|(S^2)^T|\dots|(S^m)^T|\Sigma]^T\mathbf{b} = [\mathbf{y}_{*,1}^T|\mathbf{y}_{*,2}^T|\dots|\mathbf{y}_{*,m}^T|0]^T$$
$$S^{\texttt{syst}}\mathbf{b} = \mathbf{y}$$

See illustration on Example 4.1.

Step 2: Linear dependence for $m = 2$
Let's build $S^{\texttt{syst}}$ for $m = 2$. We apply then the following linear combination:

- We multiply the n first rows by the coefficients of $\mathbf{s}_{*,2}$.
- We multiply the n following rows by the coefficient of $-\mathbf{s}_{*,1}$.
- For the column corresponding to b_{ii}, the linear combination of the rows is then equal to zero. There are only zeros in the rows of Σ for these columns.
- For the columns corresponding to b_{ij} with $i \neq j$, the linear combination on the $2n$ first rows gives a value δ_{ij}.
  - We notice that $\delta_{ij} = -\delta_{ji}$.
  - But for these columns, there is a single row of Σ having 1 in the column corresponding to b_{ij} $(i < j)$, -1 in the column of b_{ji} and 0 every where else.
  - No other row of Σ has an non-zero value under the column of b_{ij} or b_{ji}.
  - So we multiply this rows by $-\delta_{ij}$ and the linear combination gives 0 for this columns, too.

See Example 4.2 for an illustration.

Example 4.1. For instance, for $n = 3$ and $m = 2$, we have:

$$S^{\texttt{syst}} = \begin{bmatrix} s_{11} & s_{21} & s_{31} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & s_{11} & s_{21} & s_{31} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & s_{11} & s_{21} & s_{31} \\ s_{12} & s_{22} & s_{32} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & s_{12} & s_{22} & s_{32} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & s_{12} & s_{22} & s_{32} \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \end{bmatrix}$$

Example 4.2. For our example with $n = 3$, the coefficients are:

$$\{s_{12}, s_{22}, s_{32}, -s_{11}, -s_{21}, -s_{31}, s_{22}s_{11} - s_{21}s_{12}, s_{32}s_{11} - s_{31}s_{12}, s_{32}s_{21} - s_{31}s_{22}\}$$

We can easily check that this combination leads to $\mathbf{0}$.

So there is at least one linear dependence between the rows of $S^{\texttt{syst}}$. The rank of matrix $S^{\texttt{syst}}$ is at least one less than its number of lines.

Step 3: Existence conditions for $m = 2$

$\Rightarrow$ If $BS = Y$ has a solution, then $S^{\texttt{syst}}\mathbf{b} = \mathbf{y}$ has also a solution. In this case, the linear combination of the lines of matrix $S^{\texttt{syst}}$ also holds for $\mathbf{y}$.
This linear combination applied to $\mathbf{y}$ gives:

$$\mathbf{s}_{*,2}^T\mathbf{y}_{*,1} + (-\mathbf{s}_{*,1}^T)\mathbf{y}_{*,2} + 0 = 0$$

$\Leftarrow$ On the other side, if $\mathbf{s}_{*,2}^T\mathbf{y}_{*,1} + (-\mathbf{s}_{*,1}^T)\mathbf{y}_{*,2} = 0$, then applying the linear combination on $\mathbf{y}$ gives 0. The linear combination of $S^{\texttt{syst}}$ also holds for $\mathbf{y}$, so $S^{\texttt{syst}}\mathbf{b} = \mathbf{y}$ has a solution which is then also the case for $BS = Y$.

Step 4: Conditions for $m > 2$
Extending to higher value of m, for each pair i, j of the columns of S and Y, we have a solution if and only if $\mathbf{s}_{*,i}^T\mathbf{y}_{*,j} = \mathbf{s}_{*,j}^T\mathbf{y}_{*,i} = \mathbf{y}_{*,i}^T\mathbf{s}_{*,j}$. For the second, equality we used the fact that the transposed of a scalar is the scalar itself. This is equivalent to $\left[S^TY\right]_{i,j} = \left[Y^TS\right]_{i,j}$ which leads to $S^TY = Y^TS$.

QED

The last proof showed that there is at least one linear dependencies for the system $BS = Y$ and $B = B^T$ when $m = 2$. We can now try to count the number of linear dependency for higher value of m. The following theorem is an extension of Theorem 4.1.

Theorem 4.2. Let Y_m and $S_m \in \mathbb{R}^{n\times m}$, with $\frac{n+1}{2} \leq m \leq n$ and S_i full-rank. We consider the system of $n(m + \frac{n-1}{2})$ equations with n^2 unknowns resulting from the problem of finding $B \in \mathbb{R}^{n\times n}$ such that:

$$\begin{cases} BS_m &= Y_m \\ B &= B^T \end{cases}$$

This system has $\frac{m(m-1)}{2}$ equations that can be expressed as a linear combination of the other equations. So for $m > 1$, the system is not full-rank.

Proof *Step 0: Linear dependences for $m = 2$*
This has been shown above in the second proof of Theorem 4.1. By reasoning column by column, we have found one single linear combination involving all the rows.

Step 1: Linear dependences for $m = 3$
For $m = 3$, we have $S_3^{\texttt{syst}} = [S_1^T|S_2^T|S_3^T|\Sigma]^T$ (see previous demo for detail about the construction of this matrix). Thanks to step 0, we know that we have 3 linear combinations (one of each couple of columns of the matrix S). However, those linear combinations can be equivalent, as one of them could be a combination of the other two. We have to prove that those combinations are different.
We will proceed by contradiction: say that such a linear combination exists and prove that this is only possible if S is not full-rank.
We define $S_2^{\texttt{syst}} = [S_1^T|S_2^T|\Sigma]^T$:
The following facts are noted:

1. As $m = \min(i, n)$ by hypothesis, we have $n \geq 3$.
2. The dimension of $S_2^{\texttt{syst}}$ is $\frac{n(n+3)}{2} \times n^2$. The number of rows is less or equal to the number of columns.
3. As we proved that, in $S_2^{\texttt{syst}}$, there is only one linear combination involving every rows, we can take a subset of $\frac{n(n+3)}{2} - 1$ rows without losing information nor reducing the rank of the system.
4. There are at least $n + 1$ independent rows in $S_2^{\texttt{syst}}$. We have indeed n independent rows in S_1 and at least one extra independent row in S_2 because S is full-rank. Therefore, the dimension of span of its rows is at least $n + 1$.
5. The dimension of the span of the rows of S_3 is at most n.

This leads us to the following conclusion: If the span of S_3 is included in the span of $S_2^{\texttt{syst}}$, then there is only one linear combination in $S_3^{\texttt{syst}}$. Otherwise, there are 3 distinct linear combinations.

For the proof by contradiction, we assume that the span of S_3 is included in the span of $S_2^{\texttt{syst}}$. Then, for each row of S_3, there exists a linear combination of the rows of $S_2^{\texttt{syst}}$ that is equal to it.
Thanks to point 3 above, we know that we can remove one row of $S_2^{\texttt{syst}}$. We start thus by replacing the first row of S_3 (See Example 4.3).

Example 4.3. For the case $n = 3$, we have then:

$$\begin{bmatrix}
s_{13} & s_{23} & s_{33} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & s_{11} & s_{21} & s_{31} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & s_{11} & s_{21} & s_{31} \\
s_{12} & s_{22} & s_{32} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & s_{12} & s_{22} & s_{32} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & s_{12} & s_{22} & s_{32} \\
0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0
\end{bmatrix}$$

We apply then the following linear combination:

- We multiply the n first rows by the coefficients of $\mathbf{s}_{*,2}$.
- We multiply the $n + 1$-th row by $-\mathbf{s}_{1,3}$.
- We multiply the $n - 1$ following rows by the coefficient of $-\mathbf{s}_{*,1}$.
- For the column corresponding to b_{ii}, the linear combination of the rows is then equal to zero (there are only zeros in the rows of Σ for these columns).
- For the columns corresponding to b_{ij} with $i \neq j$ and $i, j \neq 1$, the linear combination on the $2n$ first rows gives a value δ_{ij}.
  - We notice that $\delta_{ij} = -\delta_{ji}$.
  - But for these columns, there is a single row of Σ having 1 in the column corresponding to b_{ij} $(i < j)$, -1 in the column of b_{ji} and 0 every where else.
  - No other row of Σ has an non-zero value under the column of b_{ij} or b_{ji}.

  - So we multiply this rows by $-\delta_{ij}$ and the linear combination gives 0 for this columns, too.

- For the columns corresponding to b_{1i} and b_{i1} with $i \neq 1$, we should also have $\delta_{1i} = -\delta_{i1}$, but the equation is not automatically satisfied. This gives us $n-1$ equations (for $i = 2, \ldots, n-1$):

$$\mathbf{s}_{1,2}\mathbf{s}_{i,3} - \mathbf{s}_{i,2}\mathbf{s}_{1,3} = \mathbf{s}_{1,2}\mathbf{s}_{i,1} - \mathbf{s}_{i,2}\mathbf{s}_{1,1} \tag{4.5}$$

Applying the same reasoning but after replacing the $n+1$-th row of $S_2^{\texttt{syst}}$ by the first row of S_3 leads to another $n+1$ equations:

$$\mathbf{s}_{1,1}\mathbf{s}_{i,3} - \mathbf{s}_{i,1}\mathbf{s}_{1,3} = \mathbf{s}_{1,1}\mathbf{s}_{i,2} - \mathbf{s}_{i,1}\mathbf{s}_{1,2} \tag{4.6}$$

Combining (4.5) and (4.6) leads to:

$$\mathbf{s}_{1,2}\mathbf{s}_{i,3} - \mathbf{s}_{i,2}\mathbf{s}_{1,3} = \mathbf{s}_{i,1}\mathbf{s}_{1,3} - \mathbf{s}_{1,1}\mathbf{s}_{i,3}$$

$$\frac{\mathbf{s}_{1,3}}{\mathbf{s}_{i,3}} = \frac{\mathbf{s}_{1,2} + \mathbf{s}_{1,1}}{\mathbf{s}_{i,2} + \mathbf{s}_{i,1}}$$

We can now apply the same process but starting with replacing successively the i-th and $n+i$-th rows of $S_2^{\texttt{syst}}$ by the i-th row of S_3, for $i = 2, ..., n$. This leads to the general equations:

$$\frac{\mathbf{s}_{i,3}}{\mathbf{s}_{i,2} + \mathbf{s}_{i,1}} = k \quad \text{for } i = 1, \ldots, n \quad \text{where } k \text{ is a constant}$$

The consequence is $\mathbf{s}_{*,3} = k(\mathbf{s}_{*,1} + \mathbf{s}_{*,2})$. S is then not full-rank which is in contradiction with our hypothesis.
So the span of S_3 is not included in the span of $S_2^{\texttt{syst}}$ and, for $m = 3$, we have 3 distinct linear combinations.

Step 2: Linear dependences for $m > 3$
The proof extends to higher value of m by applying Step 3 on every combination of 3 submatrices S_i. So each additional column of S adds one new linear combination with every other column. In general, for n and m given, we have a system of $mn + \frac{n^2-n}{2}$ equations with n^2 variables, but the rank of the matrix is only $mn + \frac{n^2-n}{2} - \frac{m^2-m}{2}$.

QED

We have tried to develop an update formula such that the new estimate of the Hessian is symmetric and satisfies multiple secant equations. We found out that this formula only exists if $Y^T S = S^T Y$. If the condition is not fulfilled, we have two different update formulas: a symmetric one closest to the multisecant matrices and a multisecant formula closest to the symmetric matrices. We've proved that the limitation comes from the fact that the addition of a second secant equation adds n new scalar equations but one of these equations is linearly independent of the other ones. We finally were able to extend the result to higher numbers of secant equations.

In the next section, we will apply the same technique but we will add one extra condition: every update formula must be a secant update, which was not the case with gPSB Sym (Formula 4.2).

4.4 Secant Update generalized PSB

Formula 4.2 (gPSB Sym) and Formula 4.3 (gPSB MS) only conditional converge to one unique form which would force us to choose between both, respectively satisfying symmetry or multi-secant equations. Next to the only conditional convergence of the formulas, there is another

problem: Formula 4.2 (gPSB Sym) doesn't satisfy any secant equation (it is, indeed, as close as possible to the set of matrices, which satisfy secant equation, but not inside. So it isn't a Secant Update formula. We can try to do better and enforce the secant update property.

For gPSB Sym (Formula 4.2), we will add the requirement that the last secant equation needs to be satisfied exactly. This will lead to Formula 4.7 (SUgPSB Sym).
For gPSB Sym (Formula 4.3), we will add the requirement that the approximate Hessian needs to be closest to the set of symmetric matrices satisfying the last secant equation exactly, instead of to be closest to the set of symmetric matrices. This will lead to Formula 4.8 (SUgPSB MS).

To do this, we will alternatively project on the set of symmetrical matrices satisfying the last secant equation and on the set of matrices satisfying multiple secant equations. For the latter, we even have two choices, which should lead to the same result, as will be explained in the next paragraphs.

We will also show that under certain conditions SUgPSB Sym and SUgPSB MS will be identical.
In Subsection 4.4.1, we define the projections we will use within this section. How we will use this projections will be explained in Subsection 4.4.2.
In Subsection 4.4.3 and Subsection 4.4.4, we make the same developments but with two different projections for the Multisecant property.

4.4.1 Projections to be used

4.4.1.1 Symmetric Secant update projection

Projection on the set of symmetrical matrices satisfying last secant equation
The projection on the set of symmetrical matrices satisfying last secant equation ($K_S \cap K_{SU}$) is PSB. But as in Section 4.1.2.3, we have a problem with the standard formulation of PSB (Formula 3.6) because it only works from already symmetric matrices. So we need a new formulation.

We will apply the method of alternating projection with:

- K_S: (4.1) for the symmetrical projection, giving ${}_j\bar{B}$
- K_{SU}: Broyden "good" (Formula 3.3) for the secant update projection, giving ${}_jB$

Let's start with ${}_0B$. We first project onto the set of symmetric matrices. For more readability and as we work within one step of the QN process, we omit the subscripts i which refers to the step of the QN process. We have:

$$\begin{aligned}
{}_0\bar{B} &= \frac{{}_0B + {}_0B^T}{2} \\
{}_0\bar{\mathbf{w}} &= \mathbf{y} - {}_0\bar{B}s \\
&= \mathbf{y} - \frac{{}_0Bs + {}_0B^Ts}{2} \\
{}_1B &= {}_0\bar{B} + \frac{{}_0\bar{\mathbf{w}}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}_i} + \frac{\mathbf{s}_0\bar{\mathbf{w}}^T}{\mathbf{s}_i^T\mathbf{s}} - \frac{{}_0\bar{\mathbf{w}}^T\mathbf{s}}{(\mathbf{s}^T\mathbf{s})^2}\mathbf{s}\mathbf{s}^T
\end{aligned}$$

We should now project on symmetric matrices again. But ${}_1B$ is already symmetric. So the alternating projection has already converged. This results to the following formula:

Formula 4.6 (PSB - Direct Form from non symmetric matrix)**.** Let $B_i \in \mathbb{R}^{n\times n}$, $\mathbf{y}_i$ and $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$ and S_i full-rank. Let B_{i+1} such that:

- $B_{i+1}\mathbf{s}_i = \mathbf{y}_i$
- $\|B_{i+1} - B_i\|_{Fr}$ is minimal

Then, B_{i+1} is given by:

$$B_{i+1} = \bar{B}_i + \frac{\bar{\mathbf{w}}_i\mathbf{s}_i^T}{\mathbf{s}_i^T\mathbf{s}_i} + \frac{\mathbf{s}_i\bar{\mathbf{w}}_i^T}{\mathbf{s}_i^T\mathbf{s}_i} - \frac{\bar{\mathbf{w}}_i^T\mathbf{s}_i}{(\mathbf{s}_i^T\mathbf{s}_i)^2}\mathbf{s}_i\mathbf{s}_i^T$$

With

- $\bar{B}_i = \frac{B_i+B_i^T}{2}$
- $\bar{\mathbf{w}}_i = \mathbf{y}_i - \bar{B}_i\mathbf{s}_i$

We note that Formula 4.6 (PSB with non-symmetrical start) can be computed as the successive application of

1. One symmetrical projection, equation (4.1)
2. One Broyden "good" update (Formula 3.3)

4.4.1.2 Multisecant projection

Projection on the set of matrices satisfying multiple secant equations
For the multisecant projection, we can reuse the Formula 4.1 (QN-LS) (projection on K_{MS}). However, we now have a new possibility: the Formula 4.3 (gPSB MS).
Indeed, the only difference is that the property "closest to a symmetric matrix" is already included in Formula 4.3 (gPSB MS). We can see this last formula as the projection onto:

$$K_{MScS} = \{A \in \mathbb{R}^{n\times n} : AS_i = Y_i \text{ and A is the closest to } K_S\}$$

We have thus the choice between two formulas:

- Formula 4.1 (QN-LS) which will be used in Section 4.4.3.
- Formula 4.3 (gPSB MS) which will be used in Section 4.4.4.

Both formulas should lead to the same target formula. We will check this at the end of Section 4.4.4.

4.4.1.3 Symmetric projection

Projection on the set of symmetric matrices
As mentioned in Section 4.4.1.1, we will split the projection on the set of symmetrical matrices satisfying last secant equation into two successive projections.
The first projection will be a projection on the set of symmetric matrices: equation (4.1).
This projection will be systematically followed by a projection on the set of symmetrical matrices satisfying last secant equation: the Broyden "good" update (Formula 3.3).

4.4.2 SU generalized PSB

As Formula 4.6 (PSB with non-symmetrical start) is in fact the direct application of two update formulas, one symmetrical projection, equation (4.1), and one Broyden "good" update (Formula 3.3). Our alternating projections will be the sequence of three projections:

1. The projection on the set of matrices satisfying multiple secant equations:
   - Formula 4.1 (QN-LS), in Section 4.4.3
   - Or 4.3 (gPSB MS), in Section 4.4.4.

   We call this step MS for multisecant.
2. The projection on symmetric matrices: equation (4.1). This is called MSS for Symmetrical projection from Multisecant formula.
3. The projection on symmetric SU matrices, Formula 4.6 (PSB with non-symmetrical start). We call this SSU for Symmetric Secant Update.

We will develop the formula using the two different projections listed in point 1 above, respectively in Section 4.4.3 and Section 4.4.4. Figure 4.2 illustrates the two sequences of projections.

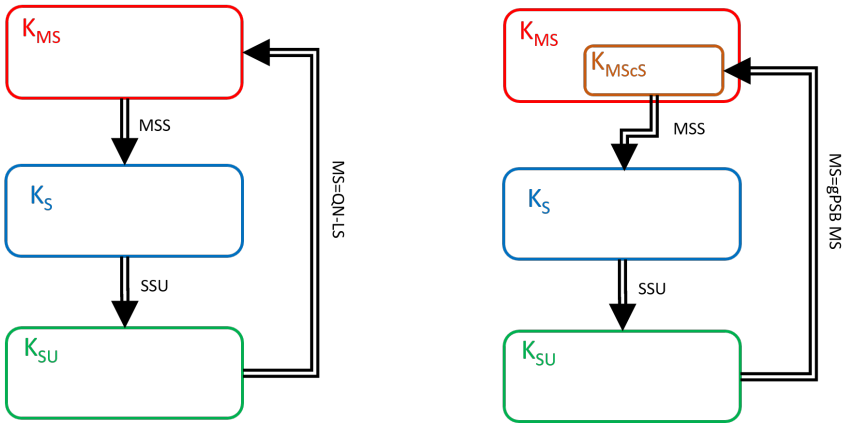

Figure 4.2:
Left: Projection sequence with QN-LS as MS projection (Section 4.4.3).
Right: Projection sequence with gPSB MS as MS projection (Section 4.4.4). K_{MScS} is a subset of K_MS.

4.4.2.1 Difficulty of the method

The main difficulty of this approach is the number of involved terms. Indeed, SSU formula applied to a k terms expression creates $4k+3$ terms. The MS step takes a k term expression and makes a $2k+1$-term or $4k+3$-term expression depending of the used formula, Formula 4.1 (QN-LS) or 4.3 (gPSB MS). For the MSS step, the number of terms is doubled. Luckily, several simplifications are possible.

In front of such a, a priori simple, distribution-simplification problem involving a lot of terms, we tried to use symbolic computer solvers, but none of the solvers we tried (Mathematica, MatLab, Python-SymPy) were able to handle the simplifications.
We finally used a mix of Excel string handling formulas, macro's and manual copy-paste to extract the wanted symbolic update formulas.

4.4.2.2 Simplifications

In the following sections, we will apply the following simplifications (and their transposed version):

- $\mathbf{s}^T\mathbf{y} = \mathbf{y}^T\mathbf{s}$ because the transposed of a scalar is the same scalar.
- $S^+S = I$ by definition of the Moore-Penrose pseudoinverse (Section 2.2.3)
- $SS^+s = s$ from Lemma 2.9
- $sSS^+ = s$ from Lemma 2.10
- $YS^+s = y$ from Lemma 2.11

4.4.3 Development of SUgPSB using Multisecant Equation

We will now apply the alternating projection method in order to find a SUgPSB. The following projections will be used:

- MS: Formula 4.1 (QN-LS) for the projection on the set on multisecant matrices, written B_{MSj}. It is the projection onto K_{MS}.
- MSS: The projection on symmetric matrices, equation (4.1), written B_{MSSj}. It is the projection onto K_S.
- SSU: The projection on symmetric SU matrices, Formula 4.6 (PSB with non-symmetrical start), written B_{SSUj}. It is the projection onto $K_S \cap K_{SU}$.

We start with a matrix B and we apply recursively the formulas:

$$MS \to MSS \to SSU \to MS \to MSS \to SSU \to MS \dots$$

This gives up:

$$\begin{aligned} B_{MS1} &= B - BSS^+ + YS^+ \\ B_{MSS1} &= \frac{B^T}{2} + \frac{B}{2} - \frac{BSS^+}{2} - \frac{(S^+)^T S^T B^T}{2} + \frac{(S^+)^T Y^T}{2} + \frac{YS^+}{2} \end{aligned}$$

In order to give an easier overview of the successive steps, we give the terms and their coefficients in Table 4.1.

We are then able to write a generic form for the j-th step. We can easily check that the application of update formulas MS, MSS and SSU on these forms gives the right following form. E.g. Applying MSS formula to equation (4.7) gives equation (4.8).

$$\begin{aligned} B_{MS_j} = &\frac{B^T}{2} - \frac{B^T SS^+}{2} + \frac{B}{2} - \frac{BSS^+}{2} - \frac{(S^+)^T S^T B^T}{2} + \frac{(S^+)^T S^T B^T SS^+}{2} - \frac{2^{j-2}-1}{2^{j-1}}(S^+)^T S^T B \\ &+ \frac{2^{j-2}-1}{2^{j-1}}(S^+)^T S^T BSS^+ + \frac{2^{j-1}-1}{2^{j-1}}(S^+)^T Y^T - \frac{2^{j-1}-1}{2^{j-1}}(S^+)^T Y^T SS^+ \\ &- \frac{1}{2^{j-1}}\frac{\mathbf{s}\mathbf{s}^T B}{\mathbf{s}^T\mathbf{s}} + \frac{1}{2^{j-1}}\frac{\mathbf{s}\mathbf{s}^T BSS^+}{\mathbf{s}^T\mathbf{s}} + \frac{1}{2^{j-1}}\frac{\mathbf{s}\mathbf{y}^T}{\mathbf{s}^T\mathbf{s}} - \frac{1}{2^{j-1}}\frac{\mathbf{s}\mathbf{y}^T SS^+}{\mathbf{s}^T\mathbf{s}} + YS^+ \end{aligned} \tag{4.7}$$

	Step 1			Step 2			Step 3			Step 4		
	MS	MSS	SSU	MS	MSS	SSU	MS	MSS	SSU	MS	MSS	SSU
B^T		$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
$\frac{B^T\mathbf{s}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}}$			$-\frac{1}{2}$		$-\frac{1}{4}$	$-\frac{1}{4}$		$-\frac{1}{8}$	$-\frac{1}{8}$		$-\frac{1}{16}$	$-\frac{1}{16}$
B^TSS^+				$-\frac{1}{2}$	$-\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{1}{2}$	$-\frac{3}{8}$	$-\frac{3}{8}$	$-\frac{1}{2}$	$-\frac{7}{16}$	$-\frac{7}{16}$
B	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
BSS^+	-1	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$
$(S^+)^TS^TB^T$		$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$
$\frac{(S^+)^TS^TB^T\mathbf{s}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}}$			$\frac{1}{2}$		$\frac{1}{4}$	$\frac{1}{4}$		$\frac{1}{8}$	$\frac{1}{8}$		$\frac{1}{16}$	$\frac{1}{16}$
$(S^+)^TS^TB^TSS^+$				$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{3}{8}$	$\frac{3}{8}$	$\frac{1}{2}$	$\frac{7}{16}$	$\frac{7}{16}$
$(S^+)^TS^TB$					$-\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{3}{8}$	$-\frac{3}{8}$	$-\frac{3}{8}$	$-\frac{7}{16}$	$-\frac{7}{16}$
$(S^+)^TS^TBSS^+$					$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{3}{8}$	$\frac{3}{8}$	$\frac{3}{8}$	$\frac{7}{16}$	$\frac{7}{16}$
$\frac{(S^+)^TS^T\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}}$					$-\frac{1}{4}$	$\frac{1}{4}$		$-\frac{1}{8}$	$\frac{3}{8}$		$-\frac{1}{16}$	$\frac{7}{16}$
$(S^+)^TS^TYS^+$					$-\frac{1}{4}$	$-\frac{1}{4}$		$-\frac{3}{8}$	$-\frac{3}{8}$		$-\frac{7}{16}$	$-\frac{7}{16}$
$(S^+)^TY^T$		$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{3}{4}$	$\frac{3}{4}$	$\frac{7}{8}$	$\frac{7}{8}$	$\frac{7}{8}$	$\frac{15}{16}$	$\frac{15}{16}$
$\frac{(S^+)^TY^T\mathbf{s}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}}$			$-\frac{1}{2}$			$-\frac{1}{2}$			$-\frac{1}{2}$			$-\frac{1}{2}$
$(S^+)^TY^TSS^+$				$-\frac{1}{2}$	$-\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{3}{4}$	$-\frac{3}{8}$	$-\frac{3}{8}$	$-\frac{7}{8}$	$-\frac{7}{16}$	$-\frac{7}{16}$
$\frac{\mathbf{s}\mathbf{s}^TB}{\mathbf{s}^T\mathbf{s}}$			$-\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{1}{8}$	$-\frac{1}{8}$	$-\frac{1}{8}$	$-\frac{1}{16}$	$-\frac{1}{16}$
$\frac{\mathbf{s}\mathbf{s}^TBSS^+}{\mathbf{s}^T\mathbf{s}}$			$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$
$\frac{\mathbf{s}\mathbf{s}^TYS^+}{\mathbf{s}^T\mathbf{s}}$			$-\frac{1}{2}$			$-\frac{1}{2}$			$-\frac{1}{2}$			$-\frac{1}{2}$
$\frac{\mathbf{s}\mathbf{y}^T}{\mathbf{s}^T\mathbf{s}}$			$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$
$\frac{\mathbf{s}\mathbf{y}^TSS^+}{\mathbf{s}^T\mathbf{s}}$				$-\frac{1}{2}$	$-\frac{1}{4}$	$\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{1}{8}$	$\frac{3}{8}$	$-\frac{1}{8}$	$-\frac{1}{16}$	$\frac{7}{16}$
$\frac{\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}}$			$\frac{1}{2}$		$\frac{1}{4}$	$\frac{1}{4}$		$\frac{1}{8}$	$\frac{1}{8}$		$\frac{1}{16}$	$\frac{1}{16}$
YS^+	1	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{3}{4}$	$\frac{3}{4}$	1	$\frac{7}{8}$	$\frac{7}{8}$	1	$\frac{15}{16}$	$\frac{15}{16}$

Table 4.1: Coefficients of the different terms for the successive steps of the sequence.

$$
\begin{aligned}
B_{MSS_j} = & \frac{B^T}{2} - \frac{1}{2^j}\frac{B^T\mathbf{s}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} - \frac{2^{j-1}-1}{2^j}B^TSS^+ + \frac{B}{2} - \frac{BSS^+}{2} - \frac{(S^+)^TS^TB^T}{2} \\
& + \frac{1}{2^j}\frac{(S^+)^TS^TB^T\mathbf{s}\mathbf{s}^T}{\mathbf{s}^t\mathbf{s}} + \frac{2^{j-1}-1}{2^j}(S^+)^TS^TB^TSS^+ - \frac{2^{j-1}-1}{2^j}(S^+)^TS^TB \\
& + \frac{2^{j-1}-1}{2^j}(S^+)^TS^TBSS^+ - \frac{1}{2^j}\frac{(S^+)^TS^T\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} - \frac{2^{j-1}-1}{2^j}(S^+)^TS^TYS^+ \\
& + \frac{2^j-1}{2^j}(S^+)^TY^T - \frac{2^{j-1}-1}{2^j}(S^+)^TY^TSS^+ - \frac{1}{2^j}\frac{\mathbf{s}\mathbf{s}^TB}{\mathbf{s}^T\mathbf{s}} \\
& + \frac{1}{2^j}\frac{\mathbf{s}\mathbf{s}^TBSS^+}{\mathbf{s}^T\mathbf{s}} + \frac{1}{2^j}\frac{\mathbf{s}\mathbf{y}^T}{\mathbf{s}^T\mathbf{s}} - \frac{1}{2^j}\frac{\mathbf{s}\mathbf{y}^TSS^+}{\mathbf{s}^T\mathbf{s}} + \frac{1}{2^j}\frac{\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} + \frac{2^j-1}{2^j}YS^+
\end{aligned} \tag{4.8}
$$

$$
\begin{aligned}
B_{SSU_j} = & \frac{B^T}{2} - \frac{1}{2^j}\frac{B^T\mathbf{s}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} - \frac{2^{j-1}-1}{2^j}B^TSS^+ + \frac{B}{2} - \frac{BSS^+}{2} - \frac{(S^+)^TS^TB^T}{2} \\
& + \frac{1}{2^j}\frac{(S^+)^TS^TB^T\mathbf{s}\mathbf{s}^T}{\mathbf{s}^t\mathbf{s}} + \frac{2^{j-1}-1}{2^j}(S^+)^TS^TB^TSS^+ - \frac{2^{j-1}-1}{2^j}(S^+)^TS^TB \\
& + \frac{2^{j-1}-1}{2^j}(S^+)^TS^TBSS^+ + \frac{2^{j-1}-1}{2^j}\frac{(S^+)^TS^T\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} - \frac{2^{j-1}-1}{2^j}(S^+)^TS^TYS^+ \\
& + \frac{2^j-1}{2^j}(S^+)^TY^T - \frac{1}{2}\frac{(S^+)^TY^T\mathbf{s}\mathbf{s}^t}{\mathbf{s}^T\mathbf{s}} - \frac{2^{j-1}-1}{2^j}(S^+)^TY^TSS^+ - \frac{1}{2^j}\frac{\mathbf{s}\mathbf{s}^TB}{\mathbf{s}^T\mathbf{s}} \\
& + \frac{1}{2^j}\frac{\mathbf{s}\mathbf{s}^TBSS^+}{\mathbf{s}^T\mathbf{s}} - \frac{1}{2}\frac{\mathbf{s}\mathbf{s}^TYS^+}{\mathbf{s}^T\mathbf{s}} + \frac{1}{2^j}\frac{\mathbf{s}\mathbf{y}^T}{\mathbf{s}^T\mathbf{s}} + \frac{2^{j-1}-1}{2^j}\frac{\mathbf{s}\mathbf{y}^TSS^+}{\mathbf{s}^T\mathbf{s}} + \frac{1}{2^j}\frac{\mathbf{y}\mathbf{s}^T}{\mathbf{s}^T\mathbf{s}} + \frac{2^j-1}{2^j}YS^+
\end{aligned} \tag{4.9}
$$

Taking the limit for $j \to \infty$ of equation (4.9), we have the SU gPBS symmetric the closest to multisecant:

Formula 4.7 (SUgPSB Sym)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$ with $m \leq n$, $\mathbf{s} = \mathbf{s}_{*,m} = (S_i)_{*,m}$, $\mathbf{y} = \mathbf{y}_{*,m} = (Y_i)_{*,m}$ and S_i full-rank. Let K_{SSUcMS} be the set of matrices B such that:

- B is symmetric
- $B\mathbf{s} = \mathbf{y}$
- B is the closest to the set $K_{MS} = \{A \in \mathbb{R}^{n\times n} : AS_i = Y_i\}$

Then, $B_{i+1} \in K_{SSUcMS}$, such that $\|B_{i+1} - B_i\|_{Fr}$ is minimal, is given by:

$$
\begin{aligned}
B_{i+1} = & \frac{B_i^T}{2} - \frac{B_i^TS_iS_i^+}{2} + \frac{B_i}{2} - \frac{B_iS_iS_i^+}{2} - \frac{(S_i^+)^TS_i^TB_i^T}{2} + \frac{(S_i^+)^TS_i^TB_i^TS_iS_i^+}{2} \\
& - \frac{(S_i^+)^TS_i^TB_i}{2} + \frac{(S_i^+)^TS_i^TB_iS_iS_i^+}{2} + \frac{(S_i^+)^TS_i^T\mathbf{y}_i\mathbf{s}_i^T}{2\mathbf{s}_i^T\mathbf{s}_i} - \frac{(S_i^+)^TS_i^TY_iS_i^+}{2} \\
& + (S_i^+)^TY_i^T - \frac{(S_i^+)^TY_i^T\mathbf{s}_i\mathbf{s}_i^T}{2\mathbf{s}_i^T\mathbf{s}_i} - \frac{(S_i^+)^TY_i^TS_iS_i^+}{2} - \frac{\mathbf{s}_i\mathbf{s}_i^TY_iS_i^+}{2\mathbf{s}_i^T\mathbf{s}_i} + \frac{\mathbf{s}_i\mathbf{y}_i^TS_iS_i^+}{2\mathbf{s}_i^T\mathbf{s}_i} + Y_iS_i^+
\end{aligned}
$$

On the other side, taking the limit for equation (4.7), we have the gPBS multisecant the closest to the symmetric SU:

Formula 4.8 (SUgPSB MS)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$ with $m \leq n$, $\mathbf{s} = \mathbf{s}_{*,m} = (S_i)_{*,m}$, $\mathbf{y} = \mathbf{y}_{*,m} = (Y_i)_{*,m}$ and S_i full-rank. Let K_{MScSSU} be the set of matrices B such that:

- $B_{i+1}S_i = Y_i$
- B_{i+1} is the closest to the set $(K_{SU} \cap K_S) = \left\{A \in \mathbb{R}^{n\times n} : A\mathbf{s} = \mathbf{y} \text{ and } A = A^T\right\}$

Then, $B_{i+1} \in K_{MScSSU}$, such that $\|B_{i+1} - B_i\|_{Fr}$ is minimal, is given by:

$$\begin{aligned} B_{i+1} = \frac{B_i^T}{2} - \frac{B_i^T S_i S_i^+}{2} + \frac{B_i}{2} - \frac{B_i S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T B_i^T}{2} + \frac{(S_i^+)^T S_i^T B_i^T S_i S_i^+}{2} \\ - \frac{(S_i^+)^T S_i^T B_i}{2} + \frac{(S_i^+)^T S_i^T B_i S_i S_i^+}{2} + (S_i^+)^T Y_i^T - (S_i^+)^T Y_i^T S_i S_i^+ + Y_i S_i^+ \end{aligned}$$

4.4.3.1 Necessary and sufficient condition of convergence

We can now verify that Formulas 4.7 (SUgPSB Sym) and 4.8 (SUgPSB MS) are equal if and only if $S_i Y_i^T = Y_i S_i^T$. This is in fact another alternative proof for Theorem 4.1.

Proof $\Leftarrow$ If B_{i+1} is symmetric and multisecant, formulas 4.7 (SUgPSB Sym) and 4.8 (SUgPSB MS) must be equal. Keeping only the differences between the formulas, we have:

$$\begin{aligned} &\frac{(S_i^+)^T S_i^T \mathbf{y}_i \mathbf{s}_i^T}{2\mathbf{s}_i^T \mathbf{s}_i} - \frac{(S_i^+)^T S_i^T Y_i S_i^+}{2} - \frac{(S_i^+)^T Y_i^T \mathbf{s}_i \mathbf{s}_i^T}{2\mathbf{s}_i^T \mathbf{s}_i} - \frac{(S_i^+)^T Y_i^T S_i S_i^+}{2} - \frac{\mathbf{s}_i \mathbf{s}_i^T Y_i S_i^+}{2\mathbf{s}_i^T \mathbf{s}_i} + \frac{\mathbf{s}_i \mathbf{y}_i^T S_i S_i^+}{2\mathbf{s}_i^T \mathbf{s}_i} \\ &= -(S_i^+)^T Y_i^T S_i S_i^+ \end{aligned}$$

As the equation must be true for every value of $\mathbf{s}$, we must have:

$$\begin{cases} \frac{(S_i^+)^T S_i^T Y_i S_i^+}{2} - \frac{(S_i^+)^T Y_i^T S_i S_i^+}{2} = -(S_i^+)^T Y_i^T S_i S_i^+ \\ (S_i^+)^T S_i^T \mathbf{y}_i \mathbf{s}_i^T - (S_i^+)^T Y_i^T \mathbf{s}_i \mathbf{s}_i^T - \mathbf{s}_i \mathbf{s}_i^T Y_i S_i^+ + \mathbf{s}_i \mathbf{y}_i^T S_i S_i^+ = 0 \end{cases}$$

The first condition leads to $Y_i^T S_i = S_i^T Y_i$ which is enough to fulfill the second condition.

$\Rightarrow$ If $Y_i^T S_i = S_i^T Y_i$, we have also $\mathbf{y}_i^T S_i = \mathbf{s}_i^T Y_i$ (the last line of $Y_i^T S_i$) and $S_i^T \mathbf{y}_i = Y_i^T \mathbf{s}_i$ (its last column). Replacing this in Formula 4.7 (SUgPSB Sym) gives Formula 4.8 (SUgPSB MS).

QED

4.4.3.2 Dual forms

Here again, we can develop the dual formulas by replacing B_i by $H_i = B_i^{-1}$, and inverting S_i and Y_i.

Formula 4.9 (ISUgPSB Sym)**.** Let $H_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$ with $m \leq n$, $\mathbf{s} = \mathbf{s}_{*,m} = (S_i)_{*,m}$, $\mathbf{y} = \mathbf{y}_{*,m} = (Y_i)_{*,m}$ and Y_i full-rank. Let $K_{ISSUcMS}$ be the set of matrices H such that:

- H is symmetric
- $H\mathbf{y} = \mathbf{s}$
- H is the closest to the set $K_{IMS} = \{A \in \mathbb{R}^{n\times n} : AY_i = S_i\}$

Then, $H_{i+1} \in K_{ISSUcMS}$, such that $\|H_{i+1} - H_i\|_{Fr}$ is minimal, is given by:

$$\begin{aligned} H_{i+1} = &\frac{H_i^T}{2} - \frac{H_i^T Y_i Y_i^+}{2} + \frac{H_i}{2} - \frac{H_i Y_i Y_i^+}{2} - \frac{(Y_i^+)^T Y_i^T H_i^T}{2} + \frac{(Y_i^+)^T Y_i^T H_i^T Y_i Y_i^+}{2} \\ &- \frac{(Y_i^+)^T Y_i^T H_i}{2} + \frac{(Y_i^+)^T Y_i^T H_i Y_i Y_i^+}{2} + \frac{(Y_i^+)^T Y_i^T \mathbf{s}_i \mathbf{y}_i^T}{2\mathbf{y}_i^T \mathbf{y}_i} - \frac{(Y_i^+)^T Y_i^T S_i Y_i^+}{2} \\ &+ (Y_i^+)^T S_i^T - \frac{(Y_i^+)^T S_i^T \mathbf{y}_i \mathbf{y}_i^T}{2\mathbf{y}_i^T \mathbf{y}_i} - \frac{(Y_i^+)^T S_i^T Y_i Y_i^+}{2} - \frac{\mathbf{y}_i \mathbf{y}_i^T S_i Y_i^+}{2\mathbf{y}_i^T \mathbf{y}_i} + \frac{\mathbf{y}_i \mathbf{s}_i^T Y_i Y_i^+}{2\mathbf{y}_i^T \mathbf{y}_i} + S_i Y_i^+ \end{aligned}$$

Formula 4.10 (ISUgPSB MS)**.** Let $H_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$ with $m \leq n$, $\mathbf{s} = \mathbf{s}_{*,m} = (S_i)_{*,m}$, $\mathbf{y} = \mathbf{y}_{*,m} = (Y_i)_{*,m}$ and Y_i full-rank. Let $K_{IMScSSU}$ be the set of matrices H such that:

- $HY_i = S_i$
- H is the closest to the set $(K_{ISU} \cap K_S) = \left\{A \in \mathbb{R}^{n\times n} : A\mathbf{y} = \mathbf{s} \text{ and } A = A^T\right\}$

Then, $H_{i+1} \in K_{MScSSU}$, such that $\|H_{i+1} - H_i\|_{Fr}$ is minimal, is given by:

$$\begin{aligned} H_{i+1} = &\frac{H_i^T}{2} - \frac{H_i^T Y_i Y_i^+}{2} + \frac{H_i}{2} - \frac{H_i Y_i Y_i^+}{2} - \frac{(Y_i^+)^T Y_i^T H_i^T}{2} + \frac{(Y_i^+)^T Y_i^T H_i^T Y_i Y_i^+}{2} \\ &- \frac{(Y_i^+)^T Y_i^T H_i}{2} + \frac{(Y_i^+)^T Y_i^T H_i Y_i Y_i^+}{2} + (Y_i^+)^T S_i^T - (Y_i^+)^T S_i^T Y_i Y_i^+ + S_i Y_i^+ \end{aligned}$$

4.4.4 Development of SUgPSB using Multisecant equation closest to Symmetric

In this second approach, we will use another projection of the MS-projection. This difference could lead to a possibly different form for the update formula. We use the now following projections:

- MS: Formula 4.3 (gPSB MS) for the projection on the set on multisecant matrices, written B_{MSj}. This differs from previous section (Section 4.4.3) as the new matrix B_{MSj} has one additional property: being the closest to the set $(K_{SU} \cap K_S) = \left\{A \in \mathbb{R}^{n\times n} : A\mathbf{s} = \mathbf{y} \text{ and } A = A^T\right\}$. It is in fact the projection onto:

$$K_{MScSSU} = \{A \in \mathbb{R}^{n\times n} : AS_i = Y_i \text{ and A is the closest to } (K_S \cap K_{SU})\}$$

- MSS: The same as in the previous section, the projection on symmetric matrices (4.1), written B_{MSSj}.
- SSU: The same as in the previous section, the projection on symmetric SU matrices, Formula 4.6 (PSB with non-symmetrical start), written B_{SSUj}.

$$B_{MS1} = B + (S^+)^T Y^T + Y S^+ - BSS^+ - (S^+)^T S^T B^T - (S^+)^T Y^T SS^+ + (S^+)^T S^T B^T SS^+$$

$$\begin{aligned} B_{MSS1} = {} & \frac{B^T}{2} + \frac{B}{2} - BSS^+ - (S^+)^T S^T B^T + \frac{(S^+)^T S^T B^T SS^+}{2} + \frac{(S^+)^T S^T BSS^+}{2} \\ & - \frac{(S^+)^T S^T Y S^+}{2} + (S^+)^T Y^T - \frac{(S^+)^T Y^T SS^+}{2} + Y S^+ \end{aligned}$$

$$\begin{aligned} B_{SSU1} = {} & \frac{B^T}{2} - \frac{B^T ss^T}{2s^T s} + \frac{B}{2} + \frac{Bss^T}{2s^T s} - BSS^+ - (S^+)^T S^T B^T + \frac{(S^+)^T S^T B^T ss^T}{2s^T s} \\ & + \frac{(S^+)^T S^T B^T SS^+}{2} - \frac{(S^+)^T S^T Bss^T}{2s^T s} + \frac{(S^+)^T S^T BSS^+}{2} + \frac{(S^+)^T S^T ys^T}{2s^T s} \\ & - \frac{(S^+)^T S^T Y S^+}{2} + (S^+)^T Y^T - \frac{(S^+)^T Y^T ss^T}{2s^T s} - \frac{(S^+)^T Y^T SS^+}{2} + \frac{ss^T B^T}{2s^T s} \\ & - \frac{ss^T B^T SS^+}{2s^T s} - \frac{ss^T B}{2s^T s} + \frac{ss^T BSS^+}{2s^T s} - \frac{ss^T Y S^+}{2s^T s} + \frac{sy^T SS^+}{2s^T s} + Y S^+ \end{aligned}$$

$$\begin{aligned} B_{MS2} = {} & \frac{B^T}{2} - \frac{B^T SS^+}{2} + \frac{B}{2} - \frac{BSS^+}{2} - \frac{(S^+)^T S^T B^T}{2} + \frac{(S^+)^T S^T B^T SS^+}{2} \\ & - \frac{(S^+)^T S^T B}{2} + \frac{(S^+)^T S^T BSS^+}{2} + (S^+)^T Y^T - (S^+)^T Y^T SS^+ + Y S^+ \end{aligned} \tag{4.10}$$

$$\begin{aligned} B_{MSS2} = {} & \frac{B^T}{2} - \frac{B^T SS^+}{2} + \frac{B}{2} - \frac{BSS^+}{2} - \frac{(S^+)^T S^T B^T}{2} + \frac{(S^+)^T S^T B^T SS^+}{2} - \frac{(S^+)^T S^T B}{2} \\ & + \frac{(S^+)^T S^T BSS^+}{2} - \frac{(S^+)^T S^T Y S^+}{2} + (S^+)^T Y^T - \frac{(S^+)^T Y^T SS^+}{2} + Y S^+ \end{aligned}$$

$$\begin{aligned} B_{SSU2} = {} & \frac{B^T}{2} - \frac{B^T SS^+}{2} + \frac{B}{2} - \frac{BSS^+}{2} - \frac{(S^+)^T S^T B^T}{2} + \frac{(S^+)^T S^T B^T SS^+}{2} \\ & - \frac{(S^+)^T S^T B}{2} + \frac{(S^+)^T S^T BSS^+}{2} + \frac{(S^+)^T S^T ys^T}{2s^T s} - \frac{(S^+)^T S^T Y S^+}{2} \\ & + (S^+)^T Y^T - \frac{(S^+)^T Y^T ss^T}{2s^T s} - \frac{(S^+)^T Y^T SS^+}{2} - \frac{ss^T Y S^+}{2s^T s} + \frac{sy^T SS^+}{2s^T s} + Y S^+ \end{aligned} \tag{4.11}$$

$$\begin{aligned} B_{MS3} = {} & \frac{B^T}{2} - \frac{B^T SS^+}{2} + \frac{B}{2} - \frac{BSS^+}{2} - \frac{(S^+)^T S^T B^T}{2} + \frac{(S^+)^T S^T B^T SS^+}{2} \\ & - \frac{(S^+)^T S^T B}{2} + \frac{(S^+)^T S^T BSS^+}{2} + (S^+)^T Y^T - (S^+)^T Y^T SS^+ + Y S^+ \end{aligned} \tag{4.12}$$

We find out that MS3 (4.10) is equal to MS2 (4.12). We already have a convergence.

The equations (4.10) and (4.11) are equal respectively to the formulas 4.8 (SUgPSB MS) and 4.7 (SUgPSB Sym), the two methods give thus the same result.
Note that the results can easily be extended to the dual formulas.

4.5 Condition of equivalence gPSB and SUgPSB

We have developed 2 pairs of formulas:

1. gPSB:
   - gPSB Sym $\in K_S = \left\{A \in \mathbb{R}^{n\times n} : A = A^T\right\}$, Formula 4.2 (gPSB Sym)
   - gPSB MS $\in K_{MS} = \{A \in \mathbb{R}^{n\times n} : AS_i = Y_i\}$, Formula 4.3 (gPSB MS)

   being the closest to each other.
2. SUgPSB:
   - SUgPSB Sym $\in (K_{SU} \cap K_S) = \left\{A \in \mathbb{R}^{n\times n} : A\mathbf{y} = \mathbf{s} \text{ and } A = A^T\right\}$, Formula 4.7 (SUgPSB Sym)
   - SUgPSB MS $\in K_{MS} = \{A \in \mathbb{R}^{n\times n} : AS_i = Y_i\}$, Formula 4.8 (SUgPSB MS)

   being the closest to each other.

We will now compare them.

First, we can note that the formulas 4.3 (gPSB MS) and 4.8 (SUgPSB MS) are identical. This is logical as the multisecant formula the closest to the SU symmetric is already a secant update. So forcing the fact that the formula should be the closest to a secant update has no impact.

We've already seen that gPSB exists if and only if $Y_i^T S_i$ is symmetric. Indeed, then Formulas 4.2 (gPSB Sym) and 4.3 (gPSB MS) are equal.
We've also seen the same for SUgPSB using Formula 4.7 (SUgPSB Sym) instead of Formula 4.2 (gPSB Sym). We now look for equality conditions for the symmetric update formulas.

Theorem 4.3. The formulas 4.2 (gPSB Sym) and 4.7 (SUgPSB Sym) are equal if and only if the last column of $S_i^T Y_i$ is equal to the last column of $Y_i^T S_i$.

Proof
$\Rightarrow$ If both formulas are equal, we compare Formula 4.2 (gPSB Sym) and Formula 4.7 (SUgPSB Sym). After simplification of identical terms, we have:

$$\begin{aligned}
0 &= \frac{(S_i^+)^T S_i^T \mathbf{y}_i \mathbf{s}_i^T}{2\mathbf{s}_i^T \mathbf{s}_i} - \frac{(S_i^+)^T Y_i^T \mathbf{s}_i \mathbf{s}_i^T}{2\mathbf{s}_i^T \mathbf{s}_i} - \frac{\mathbf{s}_i \mathbf{s}_i^T Y_i S_i^+}{2\mathbf{s}_i^T \mathbf{s}_i} + \frac{\mathbf{s}_i \mathbf{y}_i^T S_i S_i^+}{2\mathbf{s}_i^T \mathbf{s}_i} \\
&= (S_i^+)^T S_i^T \mathbf{y}_i \mathbf{s}_i^T - (S_i^+)^T Y_i^T \mathbf{s}_i \mathbf{s}_i^T - \mathbf{s}_i \mathbf{s}_i^T Y_i S_i^+ + \mathbf{s}_i \mathbf{y}_i^T S_i S_i^+ \\
&= S_i^T \mathbf{y}_i \mathbf{s}_i^T S_i - Y_i^T \mathbf{s}_i \mathbf{s}_i^T S_i - S_i^T \mathbf{s}_i \mathbf{s}_i^T Y_i + S_i^T \mathbf{s}_i \mathbf{y}_i^T S_i \\
&= (S_i^T \mathbf{y}_i - Y_i^T \mathbf{s}_i) \mathbf{s}_i^T S_i + S_i^T \mathbf{s}_i (\mathbf{y}_i^T S_i - \mathbf{s}_i^T Y_i) \\
&= (S_i^T \mathbf{y}_i - Y_i^T \mathbf{s}_i) \mathbf{s}_i^T S_i + ((S_i^T \mathbf{y}_i - Y_i^T \mathbf{s}_i) \mathbf{s}_i^T S_i)^T
\end{aligned}$$

$(S_i^T \mathbf{y}_i - Y_i^T \mathbf{s}_i)\mathbf{s}_i^T S_i$ is skew-symmetric, so by Lemma 2.4, its rank must be even.
But $(S_i^T \mathbf{y}_i - Y_i^T \mathbf{s}_i)\mathbf{s}_i^T S_i$ has the form of $\mathbf{u}\mathbf{v}^T$. By Lemma's 2.1 and 2.2, its rank is at most 1.
We deduce that its rank must be 0 which is only possible if $\mathbf{u} = \mathbf{0}$ or $\mathbf{v} = \mathbf{0}$.
$\mathbf{s}_i^T S_i$ cannot be null because by hypothesis S_i is full-rank (See section 2.4.2). So we have:

$$S_i^T \mathbf{y}_i - Y_i^T \mathbf{s}_i = \mathbf{0}$$

This is equivalent to say that the last column of $S_i^T Y_i$ is equal to the last column of $Y_i^T S_i$.

$\Leftarrow$ If the last column of $S_i^T Y_i$ is equal to the last column of $Y_i^T S_i$, we have $S_i^T \mathbf{y}_i = Y_i^T \mathbf{s}_i$ and $\mathbf{y}_i^T S_i = \mathbf{s}_i^T Y_i$. The equality is then quickly verified.

QED

The equality condition is obviously fulfilled when $Y_i^T S_i = S_i^T Y_i$. Then formulas 4.3 (gPSB MS), 4.2 (gPSB Sym) and 4.7 (SUgPSB Sym) are identical. This is then our gPSB formula.

4.6 Conclusion

In this chapter, we were looking for a symmetric update matrix that satisfy multiple secant equations. we have found the formula we were looking for. This formula only exists under specific conditions, only if $Y^T S = S^T Y$.
We have also found found some formulas that are in some ways the closest to our wanted formula. Table 3.1 was updated with these new formulas. The new summary is given in Table 4.2.
In the next chapter, we will look for other alternatives and strategies to approach the wanted formula.

Least Change on B		Least Change on H		Secant Update			Other Secants			Symmetry			
Method	Formula	Method	Formula	No	Closest to SU	Last secant	No	Closest to MS	Multiple secants	No Symmetry	Closest to Sym	Symmetric	Other properties
SR1	3.1	SR1	3.1			X	X					X	Rank 1, Not LC
BG	3.3	BB	3.5			X	X			X			
PSB	3.6	IPSB	3.7			X	X					X	
QN-LS	3.8	QN-ILS	3.9			X			X	X			
DFP	3.10	BFGS	3.12			X	X					X	Positive Definite
gPSB Sym	4.2	IgPSB Sym	4.4		X			X				X	} =(I)gPSB
gPSB MS	4.3	IgPSB MS	4.5			X			X		X		} iff
SUgPSB Sym	4.7	ISUgPSB Sym	4.9			X		X				X	} $S^TY = Y^TS$
gPSB		IgPSB				X			X			X	Iff $S^TY = Y^TS$

Table 4.2: Summary table of the existing methods discussed until now.

Chapter 5

Other strategies

In the previous chapter, we saw that the formula of pPSB only exists if $S_i^T Y_i = Y_i^T S_i$, i.e. when $S_i^T Y_i$ is symmetric. In order to find an update formula for the Hessian matrix even if the condition is not fulfilled, we developed formulas that satisfy only a limited number of the constraints but are the closest to the wanted formula.
In this chapter, we will try other strategies.

In Section 5.1, the idea is to correct Y_i (or S_i) in order to have $S_i^T Y_i = Y_i^T S_i$ and be able to apply the formula of generalized PSB (one of the Formulas 4.2, 4.3 or 4.7 as they are then all equal).

In Section 5.2, we try another strategy. As it is impossible to fulfill at the same time the symmetry and the multisecant conditions, we will only enforce the symmetry and penalize the non-fulfilling of the multisecant condition.

5.1 Correction of data

In 1983, after having proved that the generalized PSB formula only exists if and only if $S_i^T Y_i = Y_i^T S_i$, Schnabel gave two strategies to still be able to apply the formula [20]. The idea is to modify Y_i by adding δY_i such that $S_i^T (Y_i + \delta Y_i) = (Y_i + \delta Y_i)^T S_i$.

In the next subsections, we describe his two strategies. We will only make some light changes in relation to the original paper: Schnabel added new columns to matrices S_i and Y_i by appending them on the left whereas we append the matrices on the right.

5.1.1 Triangular decomposition

We observe that if $S_i^T Y_i \neq Y_i^T S_i$, then $Y_i^T S_i - S_i^T Y_i$ is a skew-symmetric matrix. We can then define L_i, a lower triangular matrix, such that:

$$Y_i^T S_i - S_i^T Y_i = -L_i + L_i^T$$

Note that the diagonal of L_i is zero.

The first strategy of Schnabel is to define δY_i such that $\delta Y_i^T S_i = -L_i^T$. This is possible if

$\delta Y_i = -S_i(S_i^T S_i)^{-1} L_i$. Indeed, we have then:

$$\begin{aligned} S_i^T(Y_i + \delta Y_i) &= S_i^T Y_i - S_i^T S_i (S_i^T S_i)^{-1} L_i = S_i^T Y_i - L_i = (Y_i^T S_i + L_i - L_i^T) - L_i = Y_i^T S_i - L_i^T \\ (Y_i + \delta Y_i)^T S_i &= Y_i S_i^T - L_i^T \left((S_i^T S_i)^{-1}\right)^T S_i^T S_i = Y_i S_i^T - L_i^T \left((S_i^T S_i)^T\right)^{-1} S_i^T S_i \\ &= Y_i S_i^T - L_i^T \left(S_i^T S_i\right)^{-1} S_i^T S_i = Y_i S_i^T - L_i^T \end{aligned}$$

We used the fact that $(A^T)^{-1} = (A^{-1})^T$.

As L_i is a lower triangular matrix with zeros on its diagonal, the last column of δY_i contains only zeros. This means that the last secant equation is still respected.

We can write it in the same form as the other formula. We call this formula "PSB - Schnabel 1" in reference to the paper of Schnabel [20]. We also give the formula for the dual problem.

Formula 5.1 (PSB-S1)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and S_i full-ranked. Let $\delta Y_i \in \mathbb{R}^{n\times m}$ such that $S_i^T(Y_i + \delta Y_i) = (Y_i + \delta Y_i)^T S_i$ and $B_{i+1} \in \mathbb{R}^{n\times n}$ such that:

- $B_{i+1} S_i = (Y_i + \delta Y_i)$
- $B_{i+1} = B_{i+1}^T$
- $\|B_{i+1} - B_i\|_{Fr}$ is minimal

We set $\delta Y_i = -S_i(S_i^T S_i)^{-1} L_i$ where L_i is the lower triangular part of $S_i^T Y_i - Y_i^T S_i$.

B_{i+1} is then given by:

$$\begin{aligned} B_{i+1} = &\frac{B_i^T}{2} - \frac{B_i^T S_i S_i^+}{2} + \frac{B_i}{2} - \frac{B_i S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T B_i^T}{2} + \frac{(S_i^+)^T S_i^T B_i^T S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T B_i}{2} \\ &+ \frac{(S_i^+)^T S_i^T B_i S_i S_i^+}{2} + (S_i^+)^T (Y_i + \delta Y_i)^T - (S_i^+)^T (Y_i + \delta Y_i)^T S_i S_i^+ + (Y_i + \delta Y_i) S_i^+ \end{aligned}$$

Formula 5.2 (IPSB-S1)**.** Let $H_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and Y_i full-ranked.
Let $\delta S_i \in \mathbb{R}^{n\times m}$ such that $Y_i^T(S_i + \delta S_i) = (S_i + \delta S_i)^T Y_i$ and $H_{i+1} \in \mathbb{R}^{n\times n}$ such that:

- $H_{i+1} Y_i = (S_i + \delta S_i)$
- $H_{i+1} = H_{i+1}^T$
- $\|H_{i+1} - H_i\|_{Fr}$ is minimal

We set $\delta S_i = -Y_i(Y_i^T Y_i)^{-1} L_i$ where L_i is the lower triangular part of $Y_i^T S_i - S_i^T Y_i$.

H_{i+1} is then given by:

$$\begin{aligned} H_{i+1} = &\frac{H_i^T}{2} - \frac{H_i^T Y_i Y_i^+}{2} + \frac{H_i}{2} - \frac{H_i Y_i Y_i^+}{2} - \frac{(Y_i^+)^T Y_i^T H_i^T}{2} \\ &+ \frac{(Y_i^+)^T Y_i^T H_i^T Y_i Y_i^+}{2} - \frac{(Y_i^+)^T Y_i^T H_i}{2} + \frac{(Y_i^+)^T Y_i^T H_i Y_i Y_i^+}{2} \\ &+ (Y_i^+)^T (S_i + \delta S_i)^T - (Y_i^+)^T (S_i + \delta S_i)^T Y_i Y_i^+ + (S_i + \delta S_i) Y_i^+ \end{aligned}$$

As the formula is built on a triangular decomposition of a matrix, we can think that the solution will depend on the order of the rows, i.e. the order in which the dimensions are defined. This is however not the case. Thanks to the following lemma, we know that $Y_i^T S_i - S_i^T Y_i$ is kept unchanged after a row swapping, so L_i is uniquely defined.

Lemma 5.1. Let U and $V \in \mathbb{R}^{n\times m}$. The value of $U^T V$ is not dependent of a row swapping applied identically on both U and V.

Proof A row swapping can be done by multiplying by a sequence of permutation matrix. Swapping the rows of U is then done thanks to: $P_k P_{k-1} \dots P_2 P_1 U$.

$$\begin{aligned}(P_k P_{k-1} \dots P_2 P_1 U)^T P_k P_{k-1} \dots P_2 P_1 V &= U^T P_1^T P_2^T \dots P_{k-1}^T P_k^T P_k P_{k-1} \dots P_2 P_1 V \\ &= U^T P_1^{-1} P_2^{-1} \dots P_{k-1}^{-1} P_k^{-1} P_k P_{k-1} \dots P_2 P_1 V \\ &= U^T V\end{aligned}$$

QED

5.1.2 Successive corrections

The second idea of Schnabel is to successively change the columns of Y_i beginning with the most recent (the columns on the right, for us) in order to respect $S_i^T Y_i = Y_i^T S_i$.
As the correction is applied recursively, we express it as an algorithm. We call this formula "PSB - Schnabel 2" in reference to the paper of Schnabel [20].

Formula 5.3 (PSB-S2)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and S_i full-ranked. Let $\delta Y_i \in \mathbb{R}^{n\times m}$ such that $S_i^T(Y_i + \delta Y_i) = (Y_i + \delta Y_i)^T S_i$ and $B_{i+1} \in \mathbb{R}^{n\times n}$ such that:

- $B_{i+1} S_i = (Y_i + \delta Y_i)$
- $B_{i+1} = B_{i+1}^T$
- $\|B_{i+1} - B_i\|_{Fr}$ is minimal

We build δY_i thanks to the following algorithm.

1. Set $[\delta Y_i]_{*,m} = \mathbf{0}$
2. For $j = m-1, \dots, 1$:
   2.1. Solve for $\mathbf{x} \in \mathbb{R}^n$

$$\arg\min_{\mathbf{x}} \|\mathbf{x}\|_2$$
$$\text{s. t.} \quad \left([Y_i]_{*,j} + \mathbf{x}\right)^T [S_i]_{*,k} = [S_i]_{*,j}^T \left([Y_i]_{*,k} + [\delta Y_i]_{*,k}\right) \quad k = m, \dots, j+1$$

   2.2. Set $[\delta Y_i]_{*,m} = \mathbf{x}$

B_{i+1} is then given by:

$$\begin{aligned}B_{i+1} = \frac{B_i^T}{2} - \frac{B_i^T S_i S_i^+}{2} + \frac{B_i}{2} - \frac{B_i S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T B_i^T}{2} + \frac{(S_i^+)^T S_i^T B_i^T S_i S_i^+}{2} - \frac{(S_i^+)^T S_i^T B_i}{2} \\ + \frac{(S_i^+)^T S_i^T B_i S_i S_i^+}{2} + (S_i^+)^T (Y_i + \delta Y_i)^T - (S_i^+)^T (Y_i + \delta Y_i)^T S_i S_i^+ + (Y_i + \delta Y_i) S_i^+\end{aligned}$$

Formula 5.4 (IPSB-S2)**.** Let $H_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$, with $m \leq n$ and Y_i full-ranked.
Let $\delta S_i \in \mathbb{R}^{n\times m}$ such that $Y_i^T(S_i + \delta S_i) = (S_i + \delta S_i)^T Y_i$ and $H_{i+1} \in \mathbb{R}^{n\times n}$ such that:

- $H_{i+1}Y_i = (S_i + \delta S_i)$
- $H_{i+1} = H_{i+1}^T$
- $\|H_{i+1} - H_i\|_{Fr}$ is minimal

We build δS_i thanks to the following algorithm.

1. Set $[\delta S_i]_{*,m} = \mathbf{0}$
2. For $j = m-1, \ldots, 1$:
   2.1. Solve for $\mathbf{x} \in \mathbb{R}^n$

$$\arg\min_{\mathbf{x}} \|\mathbf{x}\|_2$$
$$\text{s. t.} \quad \left([S_i]_{*,j} + \mathbf{x}\right)^T [Y_i]_{*,k} = [Y_i]_{*,j}^T \left([S_i]_{*,k} + [\delta S_i]_{*,k}\right) \quad k = m, \ldots, j+1$$

   2.2. Set $[\delta S_i]_{*,m} = \mathbf{x}$

H_{i+1} is then given by:

$$\begin{aligned} H_{i+1} = {} & \frac{H_i^T}{2} - \frac{H_i^T Y_i Y_i^+}{2} + \frac{H_i}{2} - \frac{H_i Y_i Y_i^+}{2} - \frac{(Y_i^+)^T Y_i^T H_i^T}{2} \\ & + \frac{(Y_i^+)^T Y_i^T H_i^T Y_i Y_i^+}{2} - \frac{(Y_i^+)^T Y_i^T H_i}{2} + \frac{(Y_i^+)^T Y_i^T H_i Y_i Y_i^+}{2} \\ & + (Y_i^+)^T (S_i + \delta S_i)^T - (Y_i^+)^T (S_i + \delta S_i)^T Y_i Y_i^+ + (S_i + \delta S_i) Y_i^+ \end{aligned}$$

The two propositions of Schnabel make possible it to use a symmetric multisecant update formula. They both keep the last secant equation unchanged. Indeed, for S2, we set $[\delta Y_i]_{*,m} = \mathbf{0}$ and, for S1, the correction is a lower triangular matrix with zeros on its diagonal, so the last column is empty.
The main difference between them is that S1 applies its correction at once while S2 apply successive correction. The correction of S2 should therefore be limited compared to the correction of S1.
The main disadvantage of S1 is the fact that the correction doesn't ensure that the correction on the most recent secants is minimal (the triangular decomposition ensure however that the correction is limited). On the other hand, S2 ensures that the correction is minimal for recent secants, but it requires $m-1$ minimizations, which increases the computation time.

We don't go further with the corrected versions of PSB and will now have a look at a new strategy which we will call penalized PSB.

5.2 Penalized PSB

We recall the system we are trying to solve.

Given $B_i \in \mathbb{R}^{n\times n}$, S_i and $Y_i \in \mathbb{R}^{n\times m}$ where $m = \min(i, n)$, we search

$$\begin{aligned} \arg\min_{B_{i+1}} \quad & \tfrac{1}{2}\left\|B_i - B_{i+1}\right\|_{Fr}^2 \\ \text{such that} \quad & B_{i+1}S_i - Y_i = 0 \\ & B_{i+1} - B_{i+1}^T = 0 \end{aligned}$$

Instead of solving this system, the second strategy is to solve a slightly different system being:

Given $B_i \in \mathbb{R}^{n\times n}$, S_i and $Y_i \in \mathbb{R}^{n\times m}$ where $m = \min(i, n)$
and ω_k a positive real scalar for $k = 1, \ldots, m$, we search

$$\begin{aligned} \arg\min_{B_{i+1}} \quad & \tfrac{1}{2}\left\|B_i - B_{i+1}\right\|_{Fr}^2 + \tfrac{1}{2}\sum_{k=1}^{m}\omega_k\left\|B_{i+1}\mathbf{s}_k - \mathbf{y}_k\right\|_2^2 \\ \text{such that} \quad & B_{i+1} - B_{i+1}^T = 0 \end{aligned}$$

The solution to this new system has been developed by Gratton, Malmedy and Toint [9], we give here their formula.

Formula 5.5 (pPSB)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$ with $m \leq n$, $\mathbf{s}_l = \mathbf{s}_{*,l} = (S_i)_{*,l}$, $\mathbf{y}_l = \mathbf{y}_{*,l} = (Y_i)_{*,l}$ and S_i full-ranked. Let ω_l positive scalar weight parameters for $l = 1, \ldots, m-1$. Let B_{i+1} such that:

- $B_{i+1} = B_{i+1}^T$
- $\left\|B_i - B_{i+1}\right\|_{Fr}^2 + \sum_{l=1}^{m}\omega_l\left\|B_{i+1}\mathbf{s}_l - \mathbf{y}_l\right\|_2^2$ is minimal

Then, B_{i+1} is given by:

$$B_{i+1} = B_i + W(2\Omega^{-1} + S^TS)^{-1}S^T + S(2\Omega^{-1} + S^TS)^{-1}W^T + S\Omega^{\frac{1}{2}}X_2\Omega^{\frac{1}{2}}S^T$$

where X_2 solves

$$(I + \Omega^{\frac{1}{2}}S^TS\Omega^{\frac{1}{2}})X_2 + X_2(I + \Omega^{\frac{1}{2}}S^TS\Omega^{\frac{1}{2}}) = -\Omega^{\frac{1}{2}}S^TW\Omega^{\frac{1}{2}}X_1 - X_1\Omega^{\frac{1}{2}}W^TS\Omega^{\frac{1}{2}}$$

with

- $W = Y - BS$
- $\Omega = \operatorname{diag}(\omega_k)$, a diagonal matrix having ω_k as elements
- $X_1 = (2I + \Omega^{\frac{1}{2}}S^TS\Omega^{\frac{1}{2}})^{-1}$

Once again, we can express the formula for the dual problem, using $H = B^{-1}$ and switching S and Y.

Formula 5.6 (IpPSB)**.** Let $H_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$ with $m \leq n$, $\mathbf{s}_l = \mathbf{s}_{*,l} = (S_i)_{*,l}$, $\mathbf{y}_l = \mathbf{y}_{*,l} = (Y_i)_{*,l}$ and Y_i full-ranked. Let ω_l positive scalar weight parameters for $l = 1, \ldots, m-1$. Let H_{i+1} such that:

- $H_{i+1} = H_{i+1}^T$
- $\|H_i - H_{i+1}\|_{Fr}^2 + \sum_{l=1}^{m} \omega_l \|H_{i+1}\mathbf{y}_l - \mathbf{s}_l\|_2^2$ is minimal

Then, H_{i+1} is given by:

$$H_{i+1} = H_i + V(2\Omega^{-1} + Y^TY)^{-1}Y^T + Y(2\Omega^{-1} + Y^TY)^{-1}V^T + Y\Omega^{\frac{1}{2}}X_2\Omega^{\frac{1}{2}}Y^T$$

where X_2 solves

$$(I + \Omega^{\frac{1}{2}}Y^TY\Omega^{\frac{1}{2}})X_2 + X_2(I + \Omega^{\frac{1}{2}}Y^TY\Omega^{\frac{1}{2}}) = -\Omega^{\frac{1}{2}}Y^TV\Omega^{\frac{1}{2}}X_1 - X_1\Omega^{\frac{1}{2}}V^TY\Omega^{\frac{1}{2}}$$

with

- $V = S - HY$
- $\Omega = \text{diag}(\omega_k)$, a diagonal matrix having ω_k as elements
- $X_1 = (2I + \Omega^{\frac{1}{2}}Y^TY\Omega^{\frac{1}{2}})^{-1}$

5.2.1 Improving penalized PSB

Formula 5.5 and Formula 5.6 have several disadvantages:

- There are two $m \times m$ matrices that have to be inverted: $(2I + \Omega^{\frac{1}{2}}S^TS\Omega^{\frac{1}{2}})$ and $(2\Omega^{-1} + S^TS)$
- For the calculation of X_2, we have to solve a Lyapunov equation.
- The formula is not a Secant Update formula

In the next section we will deal with the first problem, the matrix inversion.

5.2.2 Avoiding the matrix inversion

In order to compute a numerical value with Formula 5.5, we have to invert 2 matrices: $(2I + \Omega^{\frac{1}{2}}S^TS\Omega^{\frac{1}{2}})$ and $(2\Omega^{-1} + S^TS)$.

In the iterations of the QN process, the number of secant equations that are satisfied increases (one for the first step, two for the second step...). Between each step, we add a new column to the matrices S and Ω. We also possibly delete the first column of the matrices when they reach the maximum dimension $n \times n$. Finally, we can change the weights (elements of matrix Ω).
The particular form of those changes to the matrices makes possible to compute the value of a matrix at step $i+1$ based on the value of the matrix at step i. We will treat those modifications one by one.

5.2.2.1 Adding a column

In this subsection, we start from the matrices S and Ω and we add a column on the right side. We want to take advantage of the knowledge of S and Ω to avoid the matrix inversion.

We begin with a lemma, inspired from [12]:

Lemma 5.2. Let X be a matrix of size $n \times p$ and Y be a diagonal $n \times n$ matrix. Given $M = (X^T X + Y)^{-1}$, if we add a column $\mathbf{x}$ to X so that $\tilde{X} = [X \quad \mathbf{x}]$ and an element y to Y such that $\tilde{Y} = \text{diag}(Y, y)$, then we have,

$$\tilde{M} = (\tilde{X}^T \tilde{X} + \tilde{Y})^{-1} = \begin{bmatrix} A & \mathbf{b} \\ \mathbf{c}^T & d \end{bmatrix}$$

with:

- $A = M(I - X^T \mathbf{x}\mathbf{c}^T)$
- $\mathbf{b} = -MX^T \mathbf{x} d$
- $\mathbf{c}^T = \frac{\mathbf{x}^T X M}{\mathbf{x}^T X M X^T \mathbf{x} - y - \mathbf{x}^T \mathbf{x}}$
- $d = \frac{1}{y + \mathbf{x}^T \mathbf{x} - \mathbf{x}^T X M X^T \mathbf{x}}$

Proof We begin by developing $\tilde{M}^{-1}$.

$$\tilde{M}^{-1} = \begin{bmatrix} X^T \\ \mathbf{x}^T \end{bmatrix} \begin{bmatrix} X & \mathbf{x} \end{bmatrix} + \begin{bmatrix} Y & \mathbf{0} \\ \mathbf{0} & y \end{bmatrix} = \begin{bmatrix} X^T X + Y & X^T \mathbf{x} \\ \mathbf{x}^T X & \mathbf{x}^T \mathbf{x} + y \end{bmatrix}$$

We express now the product of $\tilde{M}^{-1}$ by $\tilde{M}$ as given in Lemma 5.2.

$$\begin{bmatrix} X^T X + Y & X^T \mathbf{x} \\ \mathbf{x}^T X & \mathbf{x}^T \mathbf{x} + y \end{bmatrix} \begin{bmatrix} A & \mathbf{b} \\ \mathbf{c}^T & d \end{bmatrix} = \begin{bmatrix} (X^T X + Y)A + X^T \mathbf{x}\mathbf{c}^T & (X^T X + Y)\mathbf{b} + X^T \mathbf{x} d \\ \mathbf{x}^T X A + (\mathbf{x}^T \mathbf{x} + y)\mathbf{c}^T & \mathbf{x}^T X \mathbf{b} + (\mathbf{x}^T \mathbf{x} + y)d \end{bmatrix}$$

We compute now each block:

$$\begin{aligned}
(X^T X + Y)A + X^T \mathbf{x}\mathbf{c}^T &= (X^T X + Y)M(I - X^T \mathbf{x}\mathbf{c}^T) + X^T \mathbf{x}\mathbf{c}^T \\
&= I - X^T \mathbf{x}\mathbf{c}^T + X^T \mathbf{x}\mathbf{c}^T \\
&= I \\
\mathbf{x}^T X A + (\mathbf{x}^T \mathbf{x} + y)\mathbf{c}^T &= \mathbf{x}^T X M (I - X^T \mathbf{x}\mathbf{c}^T) + (\mathbf{x}^T \mathbf{x} + y)\mathbf{c}^T \\
&= \mathbf{x}^T X M - (\mathbf{x}^T X M X^T \mathbf{x} - \mathbf{x}^T \mathbf{x} - y)\mathbf{c}^T \\
&= \mathbf{x}^T X M - \frac{(\mathbf{x}^T X M X^T \mathbf{x} - \mathbf{x}^T \mathbf{x} - y)\mathbf{x}^T X M}{\mathbf{x}^T X M X^T \mathbf{x} - y - \mathbf{x}^T \mathbf{x}} \\
&= \mathbf{0}^T \\
(X^T X + Y)\mathbf{b} + X^T \mathbf{x} d &= -(X^T X + Y) M X^T \mathbf{x} d + X^T \mathbf{x} d \\
&= X^T \mathbf{x} d + X^T \mathbf{x} d \\
&= \mathbf{0} \\
\mathbf{x}^T X \mathbf{b} + (\mathbf{x}^T \mathbf{x} + y)d &= -\mathbf{x}^T X M X^T \mathbf{x} d + (\mathbf{x}^T \mathbf{x} + y)d \\
&= (-\mathbf{x}^T X M X^T \mathbf{x} + \mathbf{x}^T \mathbf{x} + y)d \\
&= \frac{-\mathbf{x}^T X M X^T \mathbf{x} + \mathbf{x}^T \mathbf{x} + y}{y + \mathbf{x}^T \mathbf{x} - \mathbf{x}^T X M X^T \mathbf{x}} \\
&= 1
\end{aligned}$$

We have finally

$$\begin{bmatrix} X^TX+Y & X^T\mathbf{x} \\ \mathbf{x}^TX & \mathbf{x}^T\mathbf{x}+y \end{bmatrix} \begin{bmatrix} A & \mathbf{b} \\ \mathbf{c}^T & d \end{bmatrix} = \begin{bmatrix} I & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

QED

We easily see that $(2I + \Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}})$ and $(2\Omega^{-1} + S^T S)$ are in the form $(\tilde{Y} + \tilde{X}^T \tilde{X})$.
At the first step of the QN iteration, the dimension of X is $n \times 1$. The dimension of $(\tilde{Y} + \tilde{X}^T \tilde{X})$ is 1×1. The inversion is easy to compute.
For the next steps, we can work recursively. Applying Lemma 5.2 allow us to avoid matrix inversion (the denominator of terms $\mathbf{c}^T$ and d is a scalar).

Next to the advantage of avoiding matrix inversion, Lemma 5.2 gives another interesting advantage: the computation complexity is lower than the one for a standard matrix inversion. We recall that the complexity of computing product AB with $A \in \mathbb{R}^{p \times q}$ and $B \in \mathbb{R}^{q \times r}$ is $\mathcal{O}(pqr)$. If the dimension of X is $n \times m$, we have $\mathbf{x}^T X$ is $\mathcal{O}(mn)$.
By rightly choosing the order of the multiplications when computing the value of the elements of matrix $\tilde{M}$ in Lemma 5.2, we can restrict the calculation to vector-matrix products and no matrix-matrix product. The complexity is then limited up to $\mathcal{O}(mn)$ whereas the complexity of standard matrix inversion is $\mathcal{O}(m^{2.373})$ [23].
So, for m high enough, the formula has a second advantage, it reduces the complexity of the inversion step.

5.2.2.2 Delete a column

We start from the matrices S and Ω and we delete a column on the left. We want to take advantage of the knowledge of S and Ω to avoid the matrix inversion.

For this step, we establish a corollary of Lemma 5.2.

Lemma 5.3. Let $\tilde{M}^{-1} = (\tilde{X}^T \tilde{X})$ where $\tilde{X} = [\mathbf{x} \quad X]$ with X a $n \times p$ matrix and $\mathbf{x}$ a $n \times 1$ vector. Given

$$\tilde{M} = \begin{bmatrix} a & \mathbf{b}^T \\ \mathbf{c} & D \end{bmatrix}$$

We have $M = (X^TX)^{-1} = D - \mathbf{c}\mathbf{b}^T/a$

Proof The proof is based on the result of Lemma 5.2. Using the same reasoning as in Lemma 5.2, we find the following values for the blocks of $\tilde{M}$:

- $D = M(I - X^T\mathbf{x}\mathbf{b}^T)$
- $\mathbf{b}^T = \frac{\mathbf{x}^TXM}{\mathbf{x}^TXMX^T\mathbf{x} - \mathbf{x}^T\mathbf{x}}$
- $\mathbf{c} = -MX^T\mathbf{x}a$
- $a = \frac{1}{\mathbf{x}^T\mathbf{x} - \mathbf{x}^TXMX^T\mathbf{x}}$

We can now compute the value of $D - \mathbf{c}\mathbf{b}^T/a$.

$$\begin{aligned} D - \mathbf{c}\mathbf{b}^T/a &= M(I - X^T\mathbf{x}\mathbf{b}^T) + MX^T\mathbf{x}a\frac{\mathbf{x}^TXM}{(\mathbf{x}^TXMX^T\mathbf{x} - \mathbf{x}^T\mathbf{x})a} \\ &= M - MX^T\mathbf{x}\frac{\mathbf{x}^TXM}{\mathbf{x}^TXMX^T\mathbf{x} - \mathbf{x}^T\mathbf{x}} + MX^T\mathbf{x}\frac{\mathbf{x}^TXM}{\mathbf{x}^TXMX^T\mathbf{x} - \mathbf{x}^T\mathbf{x}} \\ &= M \end{aligned}$$

QED

Here again, as in the previous subsection, we have an easy formula to avoid the matrix inversion. This formula also combines the second advantage of previous subsection: a complexity $\mathcal{O}(mn)$.

5.2.2.3 Modify the weights

A last possible modification would be to modify the weights of the secant equations.
It would be logical to shift the weights step after step in order to apply the same weight for the latest secant equation (see following example).

Example 5.1. For our example, $n = 4$.
At the first step of the QN process, we are looking for:

$$\arg\min_{B_1} \quad \|B_0 - B_1\|_{Fr}^2 + {}_1\omega_1 \|B_1\mathbf{s}_1 - \mathbf{y}_1\|_2^2$$

At the second step, we have:

$$\arg\min_{B_2} \quad \|B_1 - B_2\|_{Fr}^2 + {}_2\omega_2 \|B_2\mathbf{s}_2 - \mathbf{y}_2\|_2^2 + {}_2\omega_1 \|B_2\mathbf{s}_1 - \mathbf{y}_1\|_2^2 2$$

We will impose ${}_2\omega_2 = {}_1\omega_1$ as it is the weight applied to the last secant equation.
The third step is:

$$\arg\min_{B_3} \quad \|B_2 - B_3\|_{Fr}^2 + {}_3\omega_3 \|B_3\mathbf{s}_3 - \mathbf{y}_3\|_2^2 + {}_3\omega_2 \|B_3\mathbf{s}_2 - \mathbf{y}_2\|_2^2 + {}_3\omega_1 \|B_3\mathbf{s}_1 - \mathbf{y}_1\|_2^2$$

We have ${}_3\omega_3 = {}_2\omega_2$ as it is the weight applied to the last secant equation, and ${}_3\omega_2 = {}_2\omega_1$ as it is the weight applied to the last but one secant equation.
For the fourth step, we have then:

$$\arg\min_{B_4} \quad \|B_3 - B_4\|_{Fr}^2 + {}_4\omega_4 \|B_4\mathbf{s}_4 - \mathbf{y}_4\|_2^2 + {}_4\omega_3 \|B_4\mathbf{s}_3 - \mathbf{y}_3\|_2^2 + {}_4\omega_2 \|B_4\mathbf{s}_2 - \mathbf{y}_2\|_2^2$$

We have ${}_4\omega_4 = {}_3\omega_3$, ${}_4\omega_3 = {}_3\omega_2$, and ${}_4\omega_2 = {}_3\omega_1$.

Term 1: $2\Omega^{-1} + S^T S$
We begin with $M = (2\Omega^{-1} + S^T S)$ where S is fixed (or has already been changed by the method developed in the previous subsections). We first note that the weights are only on the diagonal. The modification of the weights can be done thanks to Lemma 2.7.
Defining e_i a column vector with 1 as i-th element and 0 otherwise, $\omega_{i,\text{old}}$ and $\omega_{\text{i,new}}$ respectively the old and the new weight corresponding to the i-th row and column, we apply the lemma:

$$\begin{aligned}\left(M + 2(\omega_{i,\text{new}}^{-1} - \omega_{\text{i,old}}^{-1})e_i e_i^T\right)^{-1} &= M^{-1} \\ &- 2(\omega_{i,\text{new}}^{-1} - \omega_{\text{i,old}}^{-1})\frac{M^{-1}e_i e_i^T M^{-1}}{1 + 2(\omega_{i,\text{new}}^{-1} - \omega_{\text{i,old}}^{-1})e_i^T M^{-1} e_i}\end{aligned} \quad (5.1)$$

This formula makes possible to change one weight in $\mathcal{O}(m^2)$ and to avoid the matrix inversion. However, if we have to change m weights, the complexity rises to $\mathcal{O}(m^3)$.

Term 2: $2I + \Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}}$
Let's now work with $N = (2I + \Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}})$ where S is fixed. After adapting the weights, the

matrix is $N_{\text{new}} = (2I + \Omega_{\text{new}}^{\frac{1}{2}} S^T S \Omega_{\text{new}}^{\frac{1}{2}})$.
We define $\Omega_c = \Omega_{\text{new}} \Omega^{-1}$ and apply the Woodbury's formula (Lemma 2.8).

$$\begin{aligned} N_{\text{new}}^{-1} &= (2I + \Omega_{\text{new}}^{\frac{1}{2}} S^T S \Omega_{\text{new}}^{\frac{1}{2}})^{-1} \\ &= (2I + \Omega_c^{\frac{1}{2}} \Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}} \Omega_c^{\frac{1}{2}})^{-1} \\ &= \frac{1}{2} I - \frac{1}{4} \Omega_c^{\frac{1}{2}} \left((\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}})^{-1} + \frac{1}{2} \Omega_c^{\frac{1}{2}} \Omega_c^{\frac{1}{2}} \right)^{-1} \Omega_c^{\frac{1}{2}} \\ &= \frac{1}{2} I - \frac{1}{4} \Omega_c^{\frac{1}{2}} \left((\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}})^{-1} + \frac{1}{2} \Omega_c \right)^{-1} \Omega_c^{\frac{1}{2}} \end{aligned}$$

We develop:

$$\left(\frac{1}{2} \Omega_c + (\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}})^{-1} \right)^{-1} = 2\Omega_c^{-1} - 4\Omega_c^{-1} \left(\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}} + 2\Omega_c^{-1} \right)^{-1} \Omega_c^{-1}$$

Which leads to:

$$\begin{aligned} N_{\text{new}}^{-1} &= \frac{1}{2} I - \frac{1}{4} \Omega_c^{\frac{1}{2}} \left(2\Omega_c^{-1} - 4\Omega_c^{-1} \left(\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}} + 2\Omega_c^{-1} \right)^{-1} \Omega_c^{-1} \right) \Omega_c^{\frac{1}{2}} \\ &= \frac{1}{2} I - \frac{1}{4} \Omega_c^{\frac{1}{2}} 2\Omega_c^{-1} \Omega_c^{\frac{1}{2}} + \frac{1}{4} \Omega_c^{\frac{1}{2}} 4\Omega_c^{-1} \left(\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}} + 2\Omega_c^{-1} \right)^{-1} \Omega_c^{-1} \Omega_c^{\frac{1}{2}} \\ &= \frac{1}{2} I - \frac{1}{2} I + \Omega_c^{-\frac{1}{2}} \left(\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}} + 2\Omega_c^{-1} \right)^{-1} \Omega_c^{-\frac{1}{2}} \\ &= \Omega_c^{-\frac{1}{2}} \left(\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}} + 2\Omega_c^{-1} \right)^{-1} \Omega_c^{-\frac{1}{2}} \\ &= \Omega_c^{-\frac{1}{2}} \left(\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}} + 2I + 2(\Omega_c^{-1} - I) \right)^{-1} \Omega_c^{-\frac{1}{2}} \\ &= \Omega_c^{-\frac{1}{2}} \left((\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}} + 2I) + 2(\Omega \Omega_{\text{new}}^{-1} - I) \right)^{-1} \Omega_c^{-\frac{1}{2}} \end{aligned}$$

For this last expression, we see that $2(\Omega\Omega_{\text{new}}^{-1} - I)$ is a diagonal matrix and $(\Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}} + 2I)$ is known from the previous QN step. So we can reuse the equation (5.1) multiple times for every single weight correction, every non-zero value of $2(\Omega\Omega_{\text{new}}^{-1} - I)$.

So for this term again, it is possible to avoid the matrix inversion and the complexity is still $\mathcal{O}(m^3)$.

5.2.3 Open problems

In the previous subsections, we have found a method to avoid inverting the matrices. Even with this improvement, Formula 5.5 still has several disadvantages:

- The inversion of $(2I + \Omega^{\frac{1}{2}} S^T S \Omega^{\frac{1}{2}})$ and $(2\Omega^{-1} + S^T S)$ is still $\mathcal{O}(m^3)$ if the weights must be adapted at each step.
- For the calculation of X_2, we still have to solve a Lyapunov equation.
- The formula is not a Secant Update formula.

For this last reason, we will, in the next section, develop a new formula, based on the penalized PSB but enforcing the Secant Update condition.

5.3 SU penalized PSB

Using penalized PSB as defined in Section 5.2 doesn't ensure that the new approximation of the Hessian to satisfy the last secant equation. The method isn't strictly a LCSU method. By setting a very huge value for ω_i, we can numerically force the respect of this equation. However, respecting this condition analytically could lead to a different formula which can possibly solve some of the problems encountered with penalized PSB.

5.3.1 Development of SUpPSB

In this section, we will develop a version of penalized PSB that respects secant update (SUpPSB).

For more readability and to avoid to handle too many indices, we lightly change the notation in this section. We note:

- $A \equiv B_{i+1}$
- $B \equiv B_i$
- $\mathbf{s}_l$ is the k-th column of matrix S_i.
- $\mathbf{s}_m$ is the last column of S_i (the secant equation we want to respect).
- j, k the scalar coordinates within a vector or a matrix (j-th row, k-th column).
- ω_l positive scalar weight parameters for $l = 1, \ldots, m-1$

We start with the following optimization problem:

$$\begin{aligned} \arg\min_{A} \quad & \tfrac{1}{2} \|B - A\|_{Fr}^2 + \tfrac{1}{2} \sum_{l=1}^{m-1} \omega_l \|A\mathbf{s}_l - \mathbf{y}_l\|_2^2 & \\ \text{such that} \quad & A - A^T & = 0 \\ & A\mathbf{s}_m - \mathbf{y}_m & = 0 \end{aligned}$$

We take the Lagrangian of the system:

$$\mathcal{L}(A, \lambda, M) = \frac{1}{2} \|B - A\|_{Fr}^2 + \frac{1}{2} \sum_{l=1}^{m-1} \omega_l \|A\mathbf{s}_l - \mathbf{y}_l\|_2^2 + \boldsymbol{\lambda}^T (A\mathbf{s}_m - \mathbf{y}_m) + \sum_{j<k} \mu_{j,k}(A_{j,k} - A_{j,k})$$

We now take the partial derivative in function of $A_{j,k}$. We first note that:

$$\begin{aligned} \frac{\partial}{\partial A_{j,k}} \|A\mathbf{s}_l - \mathbf{y}_l\|^2 &= \frac{\partial}{\partial A_{j,k}} \left\| \begin{array}{c} \sum_i A_{1,i} [\mathbf{s}_l]_i - [\mathbf{y}_l]_1 \\ \vdots \\ \sum_i A_{n,i} [\mathbf{s}_l]_i - [\mathbf{y}_l]_m \end{array} \right\|_2^2 \\ &= \frac{\partial}{\partial A_{j,k}} \sum_k \left(\sum_i A_{k,i} [\mathbf{s}_l]_i - [\mathbf{y}_l]_i \right)^2 \\ &= \frac{\partial}{\partial A_{j,k}} \sum_j \left(\sum_k A_{j,k} [\mathbf{s}_l]_k - [\mathbf{y}_l]_k \right)^2 \\ &= 2 \left(\sum_k A_{j,k} [\mathbf{s}_l]_k - [\mathbf{y}_l]_k \right) [\mathbf{s}_l]_k \end{aligned}$$

$$\begin{aligned} &= 2\left[(A\mathbf{s}_l - \mathbf{y}_l)\right]_j \left[\mathbf{s}_l\right]_k \\ &= 2\left[(A\mathbf{s}_l - \mathbf{y}_l)\mathbf{s}_l^T\right]_{j,k} \end{aligned}$$

We find the following system:

$$\begin{cases} \dfrac{\partial \mathcal{L}}{\partial A_{j,k}} = A_{j,k} - B_{j,k} + \displaystyle\sum_{l=1}^{m-1} \omega_l \left[(A\mathbf{s}_l - \mathbf{y}_l)\mathbf{s}_l^T\right]_{j,k} + \lambda_j \left[\mathbf{s}_m\right]_k + M_{j,k} = 0 \\ \dfrac{\partial \mathcal{L}}{\partial \lambda_j} = \left[(A\mathbf{s}_m - \mathbf{y}_m)\right]_j = 0 \\ \dfrac{\partial \mathcal{L}}{\partial M_{i,j}} = A_{i,j} - A_{j,i} = 0 \end{cases}$$

With:

$$M_{i,j} = \begin{cases} \mu_{i,j} & i < j \\ -\mu_{j,i} & j < i \\ 0 & i = j \end{cases}$$

This is equivalent to:

$$\begin{cases} A - B + \displaystyle\sum_{l=1}^{m-1} \omega_l (A\mathbf{s}_l - \mathbf{y}_l)\mathbf{s}_l^T + \boldsymbol{\lambda}\mathbf{s}_m^T + M = 0 & (5.2) \\ A\mathbf{s}_m - \mathbf{y}_m = 0 & (5.3) \\ A - A^T = 0 & (5.4) \end{cases}$$

Combining (5.2) and (5.4) gives (using $M = -M^T$):

$$\begin{aligned} 0 &= A - A^T \\ &= -B + \sum_{l=1}^{m-1} \omega_l (A\mathbf{s}_l - \mathbf{y}_l)\mathbf{s}_l^T + \boldsymbol{\lambda}\mathbf{s}_m^T + M + B^T - \sum_{l=1}^{m-1} \omega_l \mathbf{s}_l(\mathbf{s}_l^T A^T - \mathbf{y}_l^T) - \mathbf{s}_m\boldsymbol{\lambda}^T - M^T \\ -2M &= -B + \sum_{l=1}^{m-1} \omega_l (A\mathbf{s}_l - \mathbf{y}_l)\mathbf{s}_l^T + \boldsymbol{\lambda}\mathbf{s}_m^T + B^T - \sum_{l=1}^{m-1} \omega_l \mathbf{s}_l(\mathbf{s}_l^T A^T - \mathbf{y}_l^T) - \mathbf{s}_m\boldsymbol{\lambda}^T \\ M &= \frac{B - B^T}{2} + \sum_{l=1}^{m-1} \omega_l \frac{\mathbf{s}_l(\mathbf{s}_l^T A^T - \mathbf{y}_l^T) - (A\mathbf{s}_l - \mathbf{y}_l)\mathbf{s}_l^T}{2} + \frac{\mathbf{s}_m\boldsymbol{\lambda}^T - \boldsymbol{\lambda}\mathbf{s}_m^T}{2} \\ &= \frac{B - B^T}{2} + \sum_{l=1}^{m-1} \omega_l \frac{\mathbf{y}_l\mathbf{s}_l^T - \mathbf{s}_l\mathbf{y}_l^T}{2} + \frac{\mathbf{s}_m\boldsymbol{\lambda}^T - \boldsymbol{\lambda}\mathbf{s}_m^T}{2} \end{aligned}$$

For the last step, we used the fact that $\mathbf{s}_l\mathbf{s}_l^T$ and A are both symmetric and that their product is commutative.

We replace then M in (5.2):

$$\begin{aligned} A &= B - \sum_{l=1}^{m-1} \omega_l (A\mathbf{s}_l - \mathbf{y}_l)\mathbf{s}_l^T - \boldsymbol{\lambda}\mathbf{s}_m^T - \frac{B - B^T}{2} - \sum_{l=1}^{m-1} \omega_l \frac{\mathbf{y}_l\mathbf{s}_l^T - \mathbf{s}_l\mathbf{y}_l^T}{2} - \frac{\mathbf{s}_m\boldsymbol{\lambda}^T - \boldsymbol{\lambda}\mathbf{s}_m^T}{2} \\ &= \bar{B} - \frac{\mathbf{s}_m\boldsymbol{\lambda}^T + \boldsymbol{\lambda}\mathbf{s}_m^T}{2} - \sum_{l=1}^{m-1} \omega_l \left(A\mathbf{s}_l\mathbf{s}_l^T - \frac{\mathbf{y}_l\mathbf{s}_l^T + \mathbf{s}_l\mathbf{y}_l^T}{2}\right) \quad (5.5) \end{aligned}$$

We used $\bar{B} = \frac{B+B^T}{2}$.
We know use equation (5.3) pre-multipled by $\mathbf{s}_m^T$:

$$\begin{aligned}
\mathbf{s}_m^T\mathbf{y}_m &= \mathbf{s}_m^T A\mathbf{s}_m \\
&= \mathbf{s}_m^T\bar{B}\mathbf{s}_m - \mathbf{s}_m^T\frac{\mathbf{s}_m\boldsymbol{\lambda}^T + \boldsymbol{\lambda}\mathbf{s}_m^T}{2}\mathbf{s}_m - \sum_{l=1}^{m-1}\omega_l\left(\mathbf{s}_m^T A\mathbf{s}_l\mathbf{s}_l^T - \mathbf{s}_m^T\frac{\mathbf{y}_l\mathbf{s}_l^T + \mathbf{s}_l\mathbf{y}_l^T}{2}\right)\mathbf{s}_m \\
&= \mathbf{s}_m^T\bar{B}\mathbf{s}_m - (\mathbf{s}_m^T\mathbf{s}_m)(\boldsymbol{\lambda}^T\mathbf{s}_m) - \sum_{l=1}^{m-1}\omega_l\left(\mathbf{y}_m^T\mathbf{s}_l\mathbf{s}_l^T\mathbf{s}_m - (\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)\right)
\end{aligned}$$

We can now extract $(\boldsymbol{\lambda}^T\mathbf{s}_m)$:

$$\begin{aligned}
(\mathbf{s}_m^T\mathbf{s}_m)(\boldsymbol{\lambda}^T\mathbf{s}_m) &= -\mathbf{s}_m^T\mathbf{y}_m + \mathbf{s}_m^T\bar{B}\mathbf{s}_m - \sum_{l=1}^{m-1}\omega_l\left(\mathbf{y}_m^T\mathbf{s}_l\mathbf{s}_l^T\mathbf{s}_m - (\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)\right) \\
(\boldsymbol{\lambda}^T\mathbf{s}_m) &= -\frac{\mathbf{s}_m^T\mathbf{y}_m}{\mathbf{s}_m^T\mathbf{s}_m} + \frac{\mathbf{s}_m^T\bar{B}\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m} - \sum_{l=1}^{m-1}\omega_l\left(\frac{\mathbf{y}_m^T\mathbf{s}_l\mathbf{s}_l^T\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m} - \frac{(\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)}{\mathbf{s}_m^T\mathbf{s}_m}\right) \\
&= -\frac{\mathbf{s}_m^T\bar{\mathbf{w}}_m}{\mathbf{s}_m^T\mathbf{s}_m} - \sum_{l=1}^{m-1}\omega_l\left(\frac{\mathbf{y}_m^T\mathbf{s}_l\mathbf{s}_l^T\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m} - \frac{(\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)}{\mathbf{s}_m^T\mathbf{s}_m}\right)
\end{aligned}$$

Equations (5.2) and (5.3) can also be combined as $A^T\mathbf{s}_m = \mathbf{y}_m$. We reuse the next result to replace $\boldsymbol{\lambda}^T\mathbf{s}_m$ and equation (5.5) to replace A.

$$\begin{aligned}
\mathbf{y}_m &= A^T\mathbf{s}_m \\
&= \bar{B}^T\mathbf{s}_m - \frac{\mathbf{s}_m\boldsymbol{\lambda}^T\mathbf{s}_m + \boldsymbol{\lambda}\mathbf{s}_m^T\mathbf{s}_m}{2} - \sum_{l=1}^{m-1}\omega_l\left(\mathbf{s}_l\mathbf{s}_l^T A^T\mathbf{s}_m - \frac{\mathbf{y}_l\mathbf{s}_l^T\mathbf{s}_m + \mathbf{s}_l\mathbf{y}_l^T\mathbf{s}_m}{2}\right) \\
&= \bar{B}^T\mathbf{s}_m - \frac{\boldsymbol{\lambda}\mathbf{s}_m^T\mathbf{s}_m}{2} + \frac{\mathbf{s}_m}{2}\left(\frac{\mathbf{s}_m^T\bar{\mathbf{w}}_m}{\mathbf{s}_m^T\mathbf{s}_m} + \sum_{l=1}^{m-1}\omega_l\left(\frac{\mathbf{y}_m^T\mathbf{s}_l\mathbf{s}_l^T\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m} - \frac{(\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)}{\mathbf{s}_m^T\mathbf{s}_m}\right)\right) \\
&\quad - \sum_{l=1}^{m-1}\omega_l\left(\mathbf{s}_l\mathbf{s}_l^T A^T\mathbf{s}_m - \frac{\mathbf{s}_l\mathbf{y}_l^T\mathbf{s}_m + \mathbf{y}_l\mathbf{s}_l^T\mathbf{s}_m}{2}\right) \\
&= \bar{B}^T\mathbf{s}_m - \frac{\boldsymbol{\lambda}\mathbf{s}_m^T\mathbf{s}_m}{2} + \frac{\mathbf{s}_m}{2}\left(\frac{\mathbf{s}_m^T\bar{\mathbf{w}}_m}{\mathbf{s}_m^T\mathbf{s}_m} + \sum_{l=1}^{m-1}\omega_l\left(\frac{\mathbf{y}_m^T\mathbf{s}_l\mathbf{s}_l^T\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m} - \frac{(\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)}{\mathbf{s}_m^T\mathbf{s}_m}\right)\right) \\
&\quad - \sum_{l=1}^{m-1}\omega_l\left(\mathbf{s}_l\mathbf{s}_l^T\mathbf{y}_m - \frac{\mathbf{s}_l\mathbf{y}_l^T\mathbf{s}_m + \mathbf{y}_l\mathbf{s}_l^T\mathbf{s}_m}{2}\right)
\end{aligned}$$

$$\begin{aligned}
\frac{\boldsymbol{\lambda}(\mathbf{s}_m^T\mathbf{s}_m)}{2} &= -\mathbf{y}_m + \bar{B}^T\mathbf{s}_m + \frac{\mathbf{s}_m}{2}\left(\frac{\mathbf{s}_m^T\bar{\mathbf{w}}_m}{\mathbf{s}_m^T\mathbf{s}_m} + \sum_{l=1}^{m-1}\omega_l\left(\frac{\mathbf{y}_m^T\mathbf{s}_l\mathbf{s}_l^T\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m} - \frac{(\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)}{\mathbf{s}_m^T\mathbf{s}_m}\right)\right) \\
&\quad - \sum_{l=1}^{m-1}\omega_l\left(\mathbf{s}_l\mathbf{s}_l^T\mathbf{y}_m - \frac{\mathbf{s}_l\mathbf{y}_l^T\mathbf{s}_m + \mathbf{y}_l\mathbf{s}_l^T\mathbf{s}_m}{2}\right)
\end{aligned}$$

$$\begin{aligned}\boldsymbol{\lambda} &= -\frac{2\mathbf{y}_m}{\mathbf{s}_m^T\mathbf{s}_m} + \frac{2\bar{B}^T\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m} + \frac{\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m}\left(\frac{\mathbf{s}_m^T\bar{\mathbf{w}}_m}{\mathbf{s}_m^T\mathbf{s}_m} + \sum_{l=1}^{m-1}\omega_l\left(\frac{\mathbf{y}_m^T\mathbf{s}_l\mathbf{s}_l^T\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m} - \frac{(\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)}{\mathbf{s}_m^T\mathbf{s}_m}\right)\right) \\ &\quad - \sum_{l=1}^{m-1}\omega_l\left(\frac{2\mathbf{s}_l\mathbf{s}_l^T\mathbf{y}_m}{\mathbf{s}_m^T\mathbf{s}_m} - \frac{\mathbf{s}_l\mathbf{y}_l^T\mathbf{s}_m + \mathbf{y}_l\mathbf{s}_l^T\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m}\right) \\ &= -\frac{2\bar{\mathbf{w}}_m}{\mathbf{s}_m^T\mathbf{s}_m} + \frac{\mathbf{s}_m\mathbf{s}_m^T\bar{\mathbf{w}}_m}{(\mathbf{s}_m^T\mathbf{s}_m)^2} \\ &\quad + \sum_{l=1}^{m-1}\omega_l\left(\frac{\mathbf{s}_m\mathbf{y}_m^T\mathbf{s}_l\mathbf{s}_l^T\mathbf{s}_m}{(\mathbf{s}_m^T\mathbf{s}_m)^2} - \frac{\mathbf{s}_m(\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)}{(\mathbf{s}_m^T\mathbf{s}_m)^2} - \frac{2\mathbf{s}_l\mathbf{s}_l^T\mathbf{y}_m}{\mathbf{s}_m^T\mathbf{s}_m} + \frac{\mathbf{s}_l\mathbf{y}_l^T\mathbf{s}_m + \mathbf{y}_l\mathbf{s}_l^T\mathbf{s}_m}{\mathbf{s}_m^T\mathbf{s}_m}\right)\end{aligned}$$

Finally, we can express equation (5.2) with only known variables, starting from form (5.5):

$$\begin{aligned}A &= \bar{B} - \frac{\mathbf{s}_m\boldsymbol{\lambda}^T}{2} - \frac{\boldsymbol{\lambda}\mathbf{s}_m^T}{2} - \sum_{l=1}^{m-1}\omega_l\left(A\mathbf{s}_l\mathbf{s}_l^T - \frac{\mathbf{y}_l\mathbf{s}_l^T + \mathbf{s}_l\mathbf{y}_l^T}{2}\right) \\ &= \bar{B} + \frac{\mathbf{s}_m\bar{\mathbf{w}}_m^T}{\mathbf{s}_m^T\mathbf{s}_m} - \frac{(\bar{\mathbf{w}}_m^T\mathbf{s}_m)\mathbf{s}_m\mathbf{s}_m^T}{2(\mathbf{s}_m^T\mathbf{s}_m)^2} + \frac{\bar{\mathbf{w}}_m\mathbf{s}_m^T}{\mathbf{s}_m^T\mathbf{s}_m} - \frac{(\mathbf{s}_m^T\bar{\mathbf{w}}_m)\mathbf{s}_m\mathbf{s}_m^T}{2(\mathbf{s}_m^T\mathbf{s}_m)^2} - \sum_{l=1}^{m-1}\omega_l\left(A\mathbf{s}_l\mathbf{s}_l^T - \frac{\mathbf{y}_l\mathbf{s}_l^T}{2} - \frac{\mathbf{s}_l\mathbf{y}_l^T}{2}\right) \\ &\quad - \sum_{l=1}^{m-1}\omega_l\left(\frac{(\mathbf{s}_m^T\mathbf{s}_l)(\mathbf{s}_l^T\mathbf{y}_m)\mathbf{s}_m\mathbf{s}_m^T}{2(\mathbf{s}_m^T\mathbf{s}_m)^2} - \frac{(\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)\mathbf{s}_m\mathbf{s}_m^T}{2(\mathbf{s}_m^T\mathbf{s}_m)^2} - \frac{(\mathbf{y}_m^T\mathbf{s}_l)\mathbf{s}_m\mathbf{s}_l^T}{\mathbf{s}_m^T\mathbf{s}_m} + \frac{(\mathbf{s}_m^T\mathbf{y}_l)\mathbf{s}_m\mathbf{s}_l^T}{2\mathbf{s}_m^T\mathbf{s}_m}\right. \\ &\qquad \left. + \frac{(\mathbf{s}_m^T\mathbf{s}_l)\mathbf{s}_m\mathbf{y}_l^T}{2\mathbf{s}_m^T\mathbf{s}_m}\right) - \sum_{l=1}^{m-1}\omega_l\left(\frac{(\mathbf{y}_m^T\mathbf{s}_l)(\mathbf{s}_l^T\mathbf{s}_m)\mathbf{s}_m\mathbf{s}_m^T}{2(\mathbf{s}_m^T\mathbf{s}_m)^2} - \frac{(\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)\mathbf{s}_m\mathbf{s}_m^T}{2(\mathbf{s}_m^T\mathbf{s}_m)^2} - \frac{(\mathbf{s}_l^T\mathbf{y}_m)\mathbf{s}_l\mathbf{s}_m^T}{\mathbf{s}_m^T\mathbf{s}_m}\right. \\ &\qquad \left. + \frac{(\mathbf{y}_l^T\mathbf{s}_m)\mathbf{s}_l\mathbf{s}_m^T}{2\mathbf{s}_m^T\mathbf{s}_m} + \frac{(\mathbf{s}_l^T\mathbf{s}_m)\mathbf{y}_l\mathbf{s}_m^T}{2\mathbf{s}_m^T\mathbf{s}_m}\right)\end{aligned}$$

Simplifying and putting terms together gives finally:

$$\begin{aligned}A\left(I + \sum_{l=1}^{m-1}\omega_l\mathbf{s}_l\mathbf{s}_l^T\right) &= \bar{B} + \frac{\mathbf{s}_m\bar{\mathbf{w}}_m^T + \bar{\mathbf{w}}_m\mathbf{s}_m^T}{\mathbf{s}_m^T\mathbf{s}_m} + \sum_{l=1}^{m-1}\omega_l\frac{\mathbf{y}_l\mathbf{s}_l^T + \mathbf{s}_l\mathbf{y}_l^T}{2} \\ &\quad - \sum_{l=1}^{m-1}\omega_l\left(\frac{\left((\mathbf{s}_m^T\mathbf{y}_l) - 2(\mathbf{y}_m^T\mathbf{s}_l)\right)\mathbf{s}_m\mathbf{s}_l^T + \left((\mathbf{y}_l^T\mathbf{s}_m) - 2(\mathbf{s}_l^T\mathbf{y}_m)\right)\mathbf{s}_l\mathbf{s}_m^T}{2\mathbf{s}_m^T\mathbf{s}_m}\right) \\ &\quad - \sum_{l=1}^{m-1}\omega_l\left(\frac{(\mathbf{s}_m^T\mathbf{s}_l)\mathbf{s}_m\mathbf{y}_l^T + (\mathbf{s}_l^T\mathbf{s}_m)\mathbf{y}_l\mathbf{s}_m^T}{2\mathbf{s}_m^T\mathbf{s}_m}\right) \\ &\quad - \sum_{l=1}^{m-1}\omega_l\left(\frac{(\mathbf{s}_m^T\mathbf{s}_l)(\mathbf{s}_l^T\mathbf{y}_m) - (\mathbf{s}_m^T\mathbf{y}_l)(\mathbf{s}_l^T\mathbf{s}_m)}{(\mathbf{s}_m^T\mathbf{s}_m)^2}\right)\mathbf{s}_m\mathbf{s}_m^T \\ &\quad - \frac{(\bar{\mathbf{w}}_m^T\mathbf{s}_m) + (\mathbf{s}_m^T\bar{\mathbf{w}}_m)}{2(\mathbf{s}_m^T\mathbf{s}_m)^2}\mathbf{s}_m\mathbf{s}_m^T\end{aligned}$$

On the right side, we have the sum of nine rank 1 matrices as each term has the form of $\mathbf{u}\mathbf{v}^T$. To see this, let's take a generic term $\sum_{l=1}^{m-1}\omega_l\mathbf{u}_l\mathbf{v}_l^T$. We define $\mathbf{w} = \sum_{l=1}^{m-1}\sqrt{\omega_l}\mathbf{u}_l$ and $\mathbf{z}^T = \sum_{l=1}^{m-1}\sqrt{\omega_l}\mathbf{v}_l^T$. We have then $\mathbf{w}\mathbf{z}^T$.
The rank is thus 1 thanks to Lemma 2.1.

On the left side, we have the term $\left(I + \sum_{l=1}^{m-1}\omega_l\mathbf{s}_l\mathbf{s}_l^T\right)$ that still should be inverted. Applying Lemma 2.7, we see that it is a regular $n \times n$ matrix if $1 + \sum_{l=1}^{m-1}\omega_l\mathbf{s}_l^T\mathbf{s}_l \neq 0$. This is always the

case as $\omega_l \geq 0$ and $\mathbf{s}_l^T \mathbf{s}_l \geq 0$. So the term can be inverted.

We can now finally express the formula for Secant Update Penalized PSB. And once again, we can work with $H_{i+1} = B_{i+1}^{-1}$ instead of B_{i+1} and switching Y_i and S_i which will give the dual formula.

Formula 5.7 (SUpPSB)**.** Let $B_i \in \mathbb{R}^{n\times n}$, Y_i and $S_i \in \mathbb{R}^{n\times m}$ with $m \leq n$, $\mathbf{s}_l = \mathbf{s}_{*,l} = (S_i)_{*,l}$, $\mathbf{y}_l = \mathbf{y}_{*,l} = (Y_i)_{*,l}$ and S_i full-ranked. Let ω_k positive scalar weight parameters for $k = 1, \ldots, m-1$. Let B_{i+1} such that:

- $B_{i+1}\mathbf{s}_m = \mathbf{y}_m$
- $B_{i+1} = B_{i+1}^T$

- $\|B_i - B_{i+1}\|_{Fr}^2 + \sum_{l=1}^{m-1} \omega_l \|B_{i+1}\mathbf{s}_l - \mathbf{y}_l\|_2^2$ is minimal

vspace-0.5cm Then, B_{i+1} is given by:

$$B_{i+1} = X\left(I + \sum_{l=1}^{m-1} \omega_l \mathbf{s}_l \mathbf{s}_l^T\right)^{-1}$$

where

$$\begin{aligned} X = \bar{B}_i &+ \frac{\mathbf{s}_m \bar{\mathbf{w}}_m^T + \bar{\mathbf{w}}_m \mathbf{s}_m^T}{\mathbf{s}_m^T \mathbf{s}_m} + \sum_{l=1}^{m-1} \omega_l \frac{\mathbf{y}_l \mathbf{s}_l^T + \mathbf{s}_l \mathbf{y}_l^T}{2} - \sum_{l=1}^{m-1} \omega_l \left(\frac{(\mathbf{s}_m^T \mathbf{s}_l)\mathbf{s}_m \mathbf{y}_l^T + (\mathbf{s}_l^T \mathbf{s}_m)\mathbf{y}_l \mathbf{s}_m^T}{2\mathbf{s}_m^T \mathbf{s}_m}\right) \\ &+ \sum_{l=1}^{m-1} \omega_l \left(\frac{\left(2(\mathbf{y}_m^T \mathbf{s}_l) - (\mathbf{s}_m^T \mathbf{y}_l)\right)\mathbf{s}_m \mathbf{s}_l^T + \left(2(\mathbf{s}_l^T \mathbf{y}_m) - (\mathbf{y}_l^T \mathbf{s}_m)\right)\mathbf{s}_l \mathbf{s}_m^T}{2\mathbf{s}_m^T \mathbf{s}_m}\right) \\ &- \sum_{l=1}^{m-1} \omega_l \left(\frac{(\mathbf{s}_m^T \mathbf{s}_l)(\mathbf{s}_l^T \mathbf{y}_m) - (\mathbf{s}_m^T \mathbf{y}_l)(\mathbf{s}_l^T \mathbf{s}_m)}{(\mathbf{s}_m^T \mathbf{s}_m)^2}\right)\mathbf{s}_m \mathbf{s}_m^T - \frac{(\bar{\mathbf{w}}_m^T \mathbf{s}_m) + (\mathbf{s}_m^T \bar{\mathbf{w}}_m)}{2(\mathbf{s}_m^T \mathbf{s}_m)^2}\mathbf{s}_m \mathbf{s}_m^T \end{aligned}$$

with $\bar{B}_i = \frac{B_i + B_i^T}{2}$ and $\bar{\mathbf{w}}_i = \mathbf{y}_i - \bar{B}_i \mathbf{s}_i$

Formula 5.8 (ISUpPSB)**.** Let $H_i \in \mathbb{R}^{n\times n}$, S_i and $Y_i \in \mathbb{R}^{n\times m}$ with $m \leq n$, $\mathbf{y}_l = \mathbf{y}_{*,l}$, $\mathbf{s}_l = \mathbf{s}_{*,l}$ and Y_i full-ranked. Let ω_k positive scalar weight parameters for $k = 1, \ldots, m-1$. Let H_{i+1} such that:

- $H_{i+1}\mathbf{y}_m = \mathbf{s}_m$
- $H_{i+1} = H_{i+1}^T$

- $\|H_i - H_{i+1}\|_{Fr}^2 + \sum_{l=1}^{m-1} \omega_l \|H_{i+1}\mathbf{y}_l - \mathbf{s}_l\|_2^2$ is minimal

Then, H_{i+1} is given by:

$$H_{i+1} = X\left(I + \sum_{l=1}^{m-1} \omega_l \mathbf{y}_l \mathbf{y}_l^T\right)^{-1}$$

where

$$\begin{aligned} X = \bar{H}_i &+ \frac{\mathbf{y}_m \bar{\mathbf{v}}_m^T + \bar{\mathbf{v}}_m \mathbf{y}_m^T}{\mathbf{y}_m^T \mathbf{y}_m} + \sum_{l=1}^{m-1} \omega_l \frac{\mathbf{s}_l \mathbf{y}_l^T + \mathbf{y}_l \mathbf{s}_l^T}{2} - \sum_{l=1}^{m-1} \omega_l \left(\frac{(\mathbf{y}_m^T \mathbf{y}_l)\mathbf{y}_m \mathbf{s}_l^T + (\mathbf{y}_l^T \mathbf{y}_m)\mathbf{s}_l \mathbf{y}_m^T}{2\mathbf{y}_m^T \mathbf{y}_m}\right) \\ &+ \sum_{l=1}^{m-1} \omega_l \left(\frac{\left(2(\mathbf{s}_m^T \mathbf{y}_l) - (\mathbf{y}_m^T \mathbf{s}_l)\right)\mathbf{y}_m \mathbf{y}_l^T + \left(2(\mathbf{y}_l^T \mathbf{s}_m) - (\mathbf{s}_l^T \mathbf{y}_m)\right)\mathbf{y}_l \mathbf{y}_m^T}{2\mathbf{y}_m^T \mathbf{y}_m}\right) \\ &- \sum_{l=1}^{m-1} \omega_l \left(\frac{(\mathbf{y}_m^T \mathbf{y}_l)(\mathbf{y}_l^T \mathbf{s}_m) - (\mathbf{y}_m^T \mathbf{s}_l)(\mathbf{y}_l^T \mathbf{y}_m)}{(\mathbf{y}_m^T \mathbf{y}_m)^2}\right)\mathbf{y}_m \mathbf{y}_m^T - \frac{(\bar{\mathbf{v}}_m^T \mathbf{y}_m) + (\mathbf{y}_m^T \bar{\mathbf{v}}_m)}{2(\mathbf{y}_m^T \mathbf{y}_m)^2}\mathbf{y}_m \mathbf{y}_m^T \end{aligned}$$

with $\bar{H}_i = \frac{H_i + H_i^T}{2}$ and $\bar{\mathbf{v}}_i = \mathbf{s}_i - \bar{H}_i \mathbf{y}_i$

5.4 Weights definition

One of the difficulties with the penalized method is that we have to define the weights. There exists plenty of methods to define them. A deep analysis of the possibilities and their results would bring us way too far out of the scope of the present study.
In order to be able to simply define the weights for our application, we develop a heuristic reasoning.

We begin with the SUpPSB (Formula 5.7). The m-th secant equation is fulfilled by definition. For the $(m-1)$-th secant equation, we have seen (see Theorem 4.2) that this equation imposes that one scalar equation is linearly dependent of the other and, in most of the cases, impossible to satisfy.
We can then impose that all but one scalar equation are satisfied and define a maximum tolerance ϵ for the last one. Let's state that $B_{i+1}s_{m-1,j} = y_{m-1,j}$ is true for $j = 1, .., n$ but $j \neq k$ and $B_{i+1}s_{m-1,k} = y_{m-1,k} + \epsilon$. The norm of $\|B_{i+1}\mathbf{s}_{m-1} - \mathbf{y}_{m-1}\|_2^2 = \epsilon^2$.
For the $(m-2)$-th secant equation, we have 2 scalar equations that cannot be respected. Let's state that:

- $B_{i+1}s_{m-2,j} = y_{m,j}$ is true for $j = 1, .., n$ but $j \neq k_1, k_2$
- $B_{i+1}s_{m-2,k_1} = y_{m-2,k_1} + \epsilon_1$
- $B_{i+1}s_{m-2,k_2} = y_{m-2,k_1-2} + \epsilon_2$

The norm of $\|B_{i+1}\mathbf{s}_{m-2} - \mathbf{y}_{m-2}\|_2^2 = \epsilon_1^2 + \epsilon_2^2 \approx 2\epsilon^2$.
The error grows linearly. It could be a good idea to let the weight decrease to compensate it: $\omega_{m-j} = \frac{C}{j}$ where C still must be defined. This has a consequence that every weighted term in the objective function of the optimization would have the same order of magnitude. The advantage of this choice is that the weights ω_j are constant for each QN step.

To define C, we have a look at $\|B_{i+1} - B_i\|_{Fr}^2 = \sum_{kj} ((B_{i+1})_{j,k} - (B_i)_{jk})^2$. Let's state we tolerate an error δ_{ij} on each term: $|(B_{i+1})_{jk} - (B_i)_{jk}| = \delta_{ij}$. We have then:

$$\|B_{i+1} - B_i\|_{Fr}^2 = \sum_{kj} ((B_{i+1})_{jk} - (B_i)_{jk})^2 \leq n^2\delta$$

where $\delta = \max \delta_{ij}$.

This can help us for the choice of the constant C:

- If $C = n^2$, respecting each secant equation is as important as having a Hessian close to the previous one.
- If $C \gg n^2$, respecting each secant equation is more important than to be close to the previous Hessian.
- On the other side, if $C \ll n^2$, the satisfying of secant equations is not so important, but the Hessian must be as close as possible to the previous one.

We now try to apply the same reasoning for pPSB (Formula 5.5).
For the first step, we have only one secant equation. We know this secant equation can be satisfied. As we will have $\|B_1\mathbf{s}_1 - \mathbf{y}_1\|_2^2 = 0$, the value of the weight parameter has no impact.
For the next steps, we have m secant equations and we know that there are $\frac{m(m-1)}{2}$ scalar equations we cannot satisfy.

We could take $\omega_1 = +\infty$, but this would lead numerically to the same result as SUpPSB (above) which we want to avoid. We will also try to keep the same kind of relation between the weights. For SUpPSB, we had two properties for $\omega_{m-j} = \frac{C}{j}$:

- The denominators follow an arithmetic progression.
- The sum of the denominators is equal to the number of scalar equations that are linearly dependent of the other ones.

Following these principles for pPSB, we have $\omega_{m-j} = \frac{C}{a+jd}$ and $\sum_{j=0}^{m-1}(a+jd) = \frac{(m-1)m}{2}$.

$$\sum_{j=0}^{m-1}(a+jd) = \frac{(m-1)m}{2}$$
$$ma + \frac{(m-1)md}{2} = \frac{(m-1)m}{2}$$

We choose to take $d = \frac{1}{m}$. We have then:

$$ma + \frac{m-1}{2} = \frac{(m-1)m}{2}$$
$$a = \frac{(m-1)^2}{2m}$$

We express now ω_{m-j}:

$$\begin{aligned}
\omega_{m-j} &= \frac{C}{a+jd} \\
&= \frac{C}{\frac{(m-1)^2}{2m} + \frac{j}{m}} \\
&= \frac{C}{\frac{(m-1)^2+2j}{2m}} \\
&= \frac{2mC}{(m-1)^2+2j}
\end{aligned}$$

We note that the weights change from QN step to QN step.
The value of C can be chosen in the same way as for SUpPSB above.

Example 5.2. This weight definition leads to the following weights:

- $m = 1$: $\omega_1 = +\infty$. But as we have already written, every value of ω_1 would lead to the same solution.
- $m = 2$: $\omega_2 = 4$, $\omega_1 = \frac{4}{3}$.
- $m = 3$: $\omega_3 = \frac{3}{2}$, $\omega_2 = 1$, $\omega_1 = \frac{3}{4}$.
- $m = 4$: $\omega_4 = \frac{8}{9}$, $\omega_3 = \frac{8}{11}$, $\omega_2 = \frac{8}{13}$, $\omega_1 = \frac{8}{15}$.

5.5 Conclusion

In this chapter, we worked on two alternative strategies to approximate the Hessian matrix by a symmetric matrix that satisfies as much as possible multiple secant equations.

We have exposed the two methods of Schnabel. The strategy there is to correct Y_i in order to force the multisecant property.

The second strategy was to penalize the non-respect of the multiple secant equations. We have first penalized every equation but the solution wasn't a LCSU formula. We have then penalized all but the last secant equation. This last formula is quite interesting, because it is a limited rank update.

We now have a lot of update formulas at our disposal. We give the summary in Table 5.1. In the next chapter, we will perform some numerical experiments in order to test their performances.

Least Change on B		Least Change on H		Secant Update				Other Secants					Symmetry			
Method	Formula	Method	Formula	No	Penalized SU	Closest to SU	Last secant	No	Corrected MS	Penalized MS	Closest to MS	Multiple secants	No Symmetry	Closest to Sym	Symmetric	Other properties
SR1	3.1	SR1	3.1				X	X							X	Rank 1, Not LC
BG	3.3	BB	3.5				X	X					X			
PSB	3.6	IPSB	3.7				X	X							X	
QN-LS	3.8	QN-ILS	3.9				X					X	X			
DFP	3.10	BFGS	3.12				X	X							X	Positive Definite
gPSB Sym	4.2	IgPSB Sym	4.4			X					X				X	=(I)gPSB
gPSB MS	4.3	IgPSB MS	4.5				X					X		X		iff
SUgPSB Sym	4.7	ISUgPSB Sym	4.9				X				X				X	$S^TY = Y^TS$
gPSB		IgPSB					X					X			X	Iff $S^TY = Y^TS$
PSB-S1	5.1	IPSB-S1	5.2				X		X						X	Triangular Corr
PSB-S2	5.3	IPSB-S2	5.4				X		X						X	Recursive Corr
pPSB	5.5	IpPSB	5.6		X					X					X	Dependent on
SUpPSB	5.7	ISUpPSB	5.8				X			X					X	choice of ω_l

Table 5.1: Summary table of the existing methods discussed until now.

Chapter 6

Numerical tests

Our objective was to develop a least squares update formula which satisfies multiple secant equations and is symmetric.
In Chapter 3, we have exposed some existing methods. None of the existing methods satisfied all the properties we wanted at once.
In Chapter 4, we have worked on generalized PSB formulas, trying to keep a maximum number of properties and being as close as possible to the other ones. We saw it is only possible to respect all the properties at once under specific conditions.
In Chapter 5, we first exposed the solutions of Schnabel [20]. The idea is to correct the data in order to be able to apply the formula of generalized PSB. In the second part of that chapter, we worked on penalized PSB. The idea there was to penalized the non-respect of secant equations instead of trying to avoid it.

We have in total 23 different formulas (see Table 5.1). In fact, we have even more formulas as penalized PSB includes the choice of, at least, one parameter (see our weight definition in Section 5.4). Each parameter variation can be seen as a new formula.

The theoretical work of developing those formulas is not a goal by itself. It is only one step of the development of a Quasi-Newton method that should be used to solve optimization problems. In this chapter, we will test our formulas to check their performances. A deep analysis of the performances of one formula is a complex study which we should do on each of the formulas we've listed in Table 5.1. This is out of the scope of the present study.
However, within the frame of our study, in order to have a first idea of the interest of the formulas we developed, we will perform a limited sequence of tests.

6.1 Testing the methods

We chose to use the testing method proposed by Moré, Burton and Garbow [16]. The idea is to test the reliability and the robustness of an algorithm by testing it on multiple problems.
The purpose of this testing is to prove that the tested algorithm works and that it works better than other algorithms for the same kind of problems. If the algorithm is only tested on a small number of tests functions, we prove its efficiency within a limited problem area. The generalization of the good performances outside the test domain is unwarranted or at least not proven.

Moré proposed a relatively large collection of test problems to optimize. Each problem can be

rewritten as:

$$\text{“Given } f_i : \mathbb{R}^n \to \mathbb{R} \text{ for } i = 1, \ldots, m \text{ with } m \geq n, \text{ solve}$$

$$\min \left\{ \sum_{i=1}^{m} f_i^2(x) : x \in \mathbb{R}^n \right\} \text{”}$$

There are in total 35 different functions in the collection.
For each problem, the global optima, and possibly local optima, are known which makes it possible to control the quality of the solution given by the tested algorithm. Moré's test procedure also includes standard calculation of the function value, its gradient or its Hessian.
The procedure makes it possible to apply a scale parameter in order to flatten down or to stretch up the function. This option can be used to check if the algorithm performs on the same way if the slopes are gentle or steep. It is like searching the highest point in plains or in mountains.
Finally, a standard starting point is defined. It is possible to change this starting point in order to estimate the performances, if we start farther away from the optimum.

We have used an existing MATLAB library[1] containing 33 of the 35 problems [27]. The two missing problems are:

- Problem 27: Brown almost-linear function
- Problem 35: Chebyquad function

6.2 Implementation

To test our formulas, we needed a collection of test functions, had to build an algorithm using the formula, and define some parameters. We list here the specifications of our experimental test procedure.

1. **Algorithm:**
   (a) We used the update formulas given or developed in the previous chapters. If we have developed an update formula for the inverse Hessian (H_i), we have used this formula instead of the update formula of B_i. For instance, we use Formulas 3.4 instead of 3.3.
   (b) If we do have not developed an update expression for H_i, we update B_i. However, in order to avoid to invert B_i, we've used the MATLAB function `gmres` which solves the system $A\mathbf{x} = \mathbf{b}$. Note that we haven't used `gmres` for the formulas directly giving H_i, this could have an impact on the results of the direct formulas (B_i) compared to the inverse formulas (H_i).
   (c) Taking $\mathbf{x}_0$ as starting point, we define $\mathbf{x}_1 = (1 - \omega)\mathbf{x}_0 - \omega H_0 \mathbf{f}(\mathbf{x}_0)$ with $\omega = 10^{-3}$.
   (d) The initialization of the Hessian is $B_0 = H_0 = I$.
   (e) To reduce numerical instability when computing S^+, we used a simple filtering technique. If $\left\|S^T S\right\|_{Fr} < 10^{-10}$, we excluded the left column of S until $\left\|S^T S\right|_{Fr} \geq 10^{-10}$. Y is then corrected to have the same number of columns as S. The filter applies of course to S and Y exchanged if we need to compute Y^+.
   This filter is not optimized. For example, if the last two secants are linearly dependent, the filter will only keep the very last secant and forget all previous points. There exists more complex filters that exclude only the linearly dependent column of the matrix. The use of such filters is a possible improvement to our algorithm.

[1] We had however to fix a small bug within the library.

(f) Based on the hypothesis that $\mathbf{f}(\mathbf{x}_i)$ is very costly to compute, we've tried to minimize the number of function evaluations. So we did not implement a step size optimization (such as a line search) in our algorithm. This has a clear impact on the performance of the algorithm. A quick comparison of our results with [28] (which uses line search BFGS and SR1) shows that our algorithm diverges more often and reaches the optimum in more iterations when it converges.

2. **Stopping criteria:**

   (a) OR When the steps are too small $\frac{\|\mathbf{x}_{i+1}-\mathbf{x}_i\|_2}{\|\mathbf{x}_i\|_2} \leq 10^{-6}$.

   (b) OR When the gradient is too small $\|\mathbf{f}(\mathbf{x}_i)\|_2 \leq 10^{-6}$.

   (c) OR When maximum number of iterations is reached $i \geq 5000$.

3. **Test functions:**

   (a) We used the standard starting point of the test functions.

   (b) We used no scale factor for the test functions.

4. **Remarks on tested formulas:**

   (a) For QN-LS, we used Formula 4.1 instead of Formula 3.8. For QN-ILS, we use the dual form of Formula 4.1.

   (b) For the (SU) penalized PSB formula, we used the weight definition we developed in Section 5.4. We have considered three choices for the parameter C: 10^3, 1 and 10^{-3}. We will refer to these formulas by adding 3, 0 or -3 after the abbreviation we used until now. For instance, the Inverse Secant Update penalized PSB with weight constant $C = 10^{-3}$ is called ISUpPSB -3; the penalized PSB with weight constant $C = 1$ is called pPSB 0.

   (c) For the Formula 5.5 and Formula 5.6, we did not implement the methods developed in Chapter 5, to avoid matrix inversion.

   (d) For the resolution of the Lyapunov equations for pPSB, we used the MATLAB function `lyap`. In some cases, we got an error telling that the solution didn ot exist or was not unique. This happened when some elements of the matrices took infinite values, but also when the eigenvalues α_i of A and β_j of B didn't satisfy $\alpha_i + \beta_j \neq 0$ for all pairs (i, j). We did not investigate this in detail, we simply catch the exception and stop the algorithm for this combination of tested formula and test problem.

6.3 Comparing methods

In total, we have tested 31 formulas (23 basic formulas but 3 parametric variations for four of them) on 33 problems. This is $31 \times 33 = 1023$ different results to analyze and $31 \times 30 \times 33 = 30690$ pairs of formulas applied to a test problem to compare. This is way too much to be done without an aggregation method.

To aggregate the results, we will find inspiration from the the sport world vocabulary. We will compare the formulas in a pairwise manner which we will call a "match". The match consists of multiple "rounds", each being the comparison of the results between the two tested formulas on one single test. We will then aggregate the results of the matches into something similar to a "championship table".

6.3.1 Round

To decide which of the two formulas wins a match, we first have to validate the solution found by the algorithm. Did the algorithm converge to the solution, did it diverge or did it stop too far away from the optimum? If the algorithm has converged and if there are multiple known local optima, we should also determine which optimum has been reached.

The divergence is easy to determine. For the convergence, it's a little more difficult. We have considered that the optimum is reached:

- If the value of the function is smaller than 1 for known optimum equal to zero.

$$|g(\mathbf{x})| \leq 1 \text{ if } g(\mathbf{x}_{\text{opt}}) = 0$$

- If the value of the known optimum is different to zero, a relative error smaller than 1.

$$\left|\frac{g(\mathbf{x}) - g(\mathbf{x}_{\text{opt}})}{g(\mathbf{x}_{\text{opt}})}\right| \leq 1 \text{ if } g(\mathbf{x}_{\text{opt}}) \neq 0$$

We know that these conditions are quite loose. Due to the form of the test functions and, possibly, to the limitation not to use step optimization like line search, we do not observe convergence very frequently. Keeping a loose criterion is a way to consider that the algorithm has lead us close enough to the optimum for a given application, so we can stop; or to consider that the algorithm has lead us close enough such that we can use another more costly optimization algorithm for the last steps.

As in our hypothesis, we stated that the most costly part of the algorithm is the calculation of the value of the function. We compare the efficiency of two formulas only on the number of function calls.
The "victory" conditions are then:

1. If none of the formulas converges (following the conditions mentioned above), no one wins the round.
2. If only one of the formulas converges, it wins the round.
3. If both converge to different optima, no one wins the round.
4. If both converge to the same optimum, the one with the smallest number of iterations wins the round. If the number of iterations is equal, it's a tie, no one wins.

6.3.2 Match

A match is the result of the 33 rounds between two formulas. We simply count the number of rounds won by each formula.

6.3.3 Championship table

To aggregate the matches, we give the formula 2 points for each won match and 1 point in case of tie.
It is then possible to sort the formulas on the number of earned points. In case of equal number of points, it is possible to use the number of won matches to distinguish dead heats.

6.4 Results

We give the resuts in Tabel 6.1 and have represented them in Figure 6.1. On the horizontal axis, we have the number of points each formula's earned. On the vertical axis, it is the total number of row victories. The link between the formula abbreviation, its properties, and frame wherein we have given its expression are given in Table 5.1.
The general correlation and linear trend observed is logical, as winning a lot of rounds ensures to win more matches and so earn more points.

We will analyze the results first generally and then with a closer look at the generalized PSB formulas and the penalized PSB formulas.

6.4.1 Generalities

We see a convergence in only about 20% of the cases. In comparison, Zhang and Xu [28] have tested SR1 and BFGS for 19 of the problems, only adding a line search to the algorithm, and saw no divergences.
With our method, we have respectively 21 and 15 convergence for SR1 and BFGS. The difference cannot only be explained by a judicious choice of the test functions by Zhang and Xu as they have a convergence for Powell badly scale function and Brown badly scale function that we do not have.
The use of line search clearly is an interesting feature that we miss, which has a negative impact on our results.

We would like to draw attention to the fact that the functions collection of Moré contains almost only difficult functions with steep slopes or almost flat parts. The fact that a formula does not solve a lot of Moré's problems does not mean that the formula will be totally useless. It only shows that the reliability and the robustness of the method is not very good. It is however possible that the algorithm exhibits very good performances in some specific application.

Another observation is that classical formulas (SR1, Broyden, PSB, QN-LS...) score very well. There are 6 classical methods in our top 10. It is interesting to see that the simplest method (SR1) is the one that overrules the other ones.

Comparing the direct or dual version of the formulas, we observe that we have in general better results for the direct variant. However, we recall that Broyden called his second formula the "bad" one. This was based on the results and that later results showed that the performances are problem related. So we won't conclude that the direct formulas are better in all cases.
Moreover, we have a small difference between our direct (B_i) and our inverse (H_i) algorithms. Indeed, we used `gmres` for the direct formulas and matrix multiplication for the inverse formulas. This difference within the calculation of the iteration points could have had an impact on the results.

6.4.2 Generalized PSB

In our examples, we have the same hierarchy between the symmetric, multisecant, and secant update symmetric version of the generalized PSB formula for the direct and the inverse formulas. The multisecant version seems to be better than the symmetric version. however, the secant update symmetric formula, (I)SUgPSB, dominates the other two.
We also note that SUpPSB scores quite well compared to the classical formula.

Algo	Won	Tide	Score	Rounds won
SR1	30	0	60	481
SUgPSB Sym	28	0	56	355
QN-ILS	27	0	54	309
gPSB MS	25	1	51	304
PSB-S2	25	1	51	292
QN-LS	25	0	50	281
BFGS	24	1	49	272
gPSB Sym	20	4	44	255
DFP	20	3	43	321
BG	20	3	43	281
PSB	20	2	42	283
PSB-S1	19	3	41	259
ISUgPSB Sym	18	4	40	306
IPSB	15	2	32	194
BB	15	1	31	170
IgPSB MS	13	3	29	172
IPSB-S2	13	3	29	172
pPSB 0	12	3	27	197
pPSB 3	11	1	23	171
IgPSB Sym	11	1	23	127
IPBS-S1	11	0	22	122
SUpPSB -3	8	1	17	101
IpPSB 3	6	3	15	93
ISUpPSB 0	4	4	12	65
SUpPSB 3	4	4	12	63
SUpPSB 0	5	2	12	53
ISUpPSB 3	4	2	10	41
pPSB -3	2	1	5	46
ISUpPSB -3	1	2	4	16
IpPSB -3	1	1	3	38
IpPSB 0	0	0	0	43

Table 6.1: Results of the tests

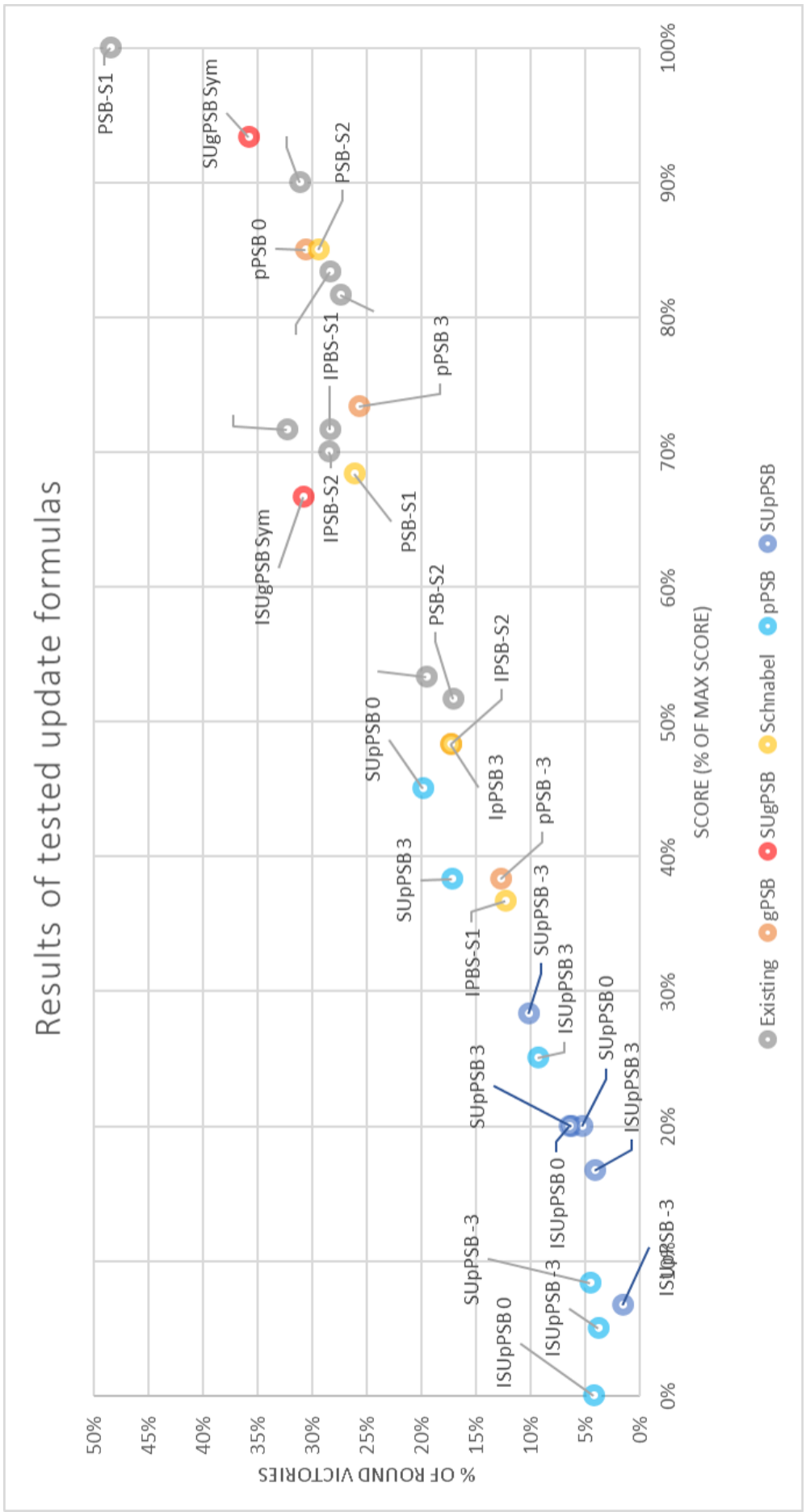

Figure 6.1: Results of the tests

This is encouraging. It seems to corroborate the hypothesis at the origin of this study: developing a method that combines the advantages of symmetric methods and multisecant methods should prove beneficial.

6.4.3 Penalized PSB

Both (I)pPSB and (I)SUpPSB score badly. We could conclude that the method is not adequate for solving problems. We recall however that we didn't execute a deep analysis of the best way to define the weights. More work in this direction is needed to completely discard this method or to find the optimal way to define the weights.
We are already able to observe that the use of a greater weight constant C seems to have a good influence on the final result. This could be an interesting first step for the analysis.

Despite the bad score, we see that the results of pPSB and SUpPSB are comparable. Recalling that pPSB is quite difficult to compute, due to the Lyapunov equation and the matrix inversions, is a first argument in favor of the use of SUpPSB instead of pPSB. If two methods seem to give the same results, it's better to use the one that is the simplest to run.

6.5 Conclusion

This chapter was a first step in the validation of the formulas we have listed, improved or developed in the previous three chapters. Even if our approach was mainly theoretical, the development of optimization methods has no sense if the methods are not used on practical problems.
The formulas have been used within a simple algorithm and tested on 33 different optimization problems.

The results for the generalized PSB are quite good. The improvement we tried by developing a secant update version of the formulas shows encouraging results. The SU formula seems to be quite competitive.

The results on penalized PSB are not so good compared to the other methods. This should lead to a deeper analysis in order to confirm this result or to improve the performance.
Regardless the general bad results compared to other methods, the Secant Update version we have developed keeps being interesting as it performs in a comparable way to the original method, while being much easier to compute.

These first results open multiple doors for further studies. We will give some of them in our final conclusion in the next chapter.

Chapter 7

Conclusions and possible way-ahead

In Chapter 1, we shortly explained the principle of Quasi-Newton optimization methods. We exposed several possible properties of the estimated Hessian, in particular, the secant update property, the multisecant proprety, and symmetry. We stated that we were looking for an update formula that was, at the same time, symmetric and multisecant. The problem has been fully mathematically stated at the end of Chapter 2.

In Chapter 3, we exposed existing methods that exhibit some of the wanted properties. We also systematically exposed the dual form of the formula, the update of the inverse of the Hessian instead of the Hessian itself.

In Chapter 4, we developed a generalized PSB formula that is a Least Squares Secant Update Symmetric Multisecant Update formula. We proved that this formula only exists under specific conditions. If this condition is not fulfilled, the formula splits into a multisecant and a symmetric version.
We were able to explain the phenomenon that leads to this impossibility which we found nowhere in the literature.
Finally, a Secant Update version of the symmetric formula we found before, has been developed. If some existing results are close enough to Formula 4.2 and Formula 4.3 to make us think that the results were already known (but not published), we found nowhere indications of the development of a SUgPSB Sym (Formula 4.7). We conclude that this is, if not a new, at least a less explored track.

Chapter 5 contains some alternative to the generalized PSB. The first idea was to correct the value in order to being able to apply the gPSB. We simply exposed these strategies.
The second idea was to penalize the non-respect of secant equation instead of imposing them. This leads to the formula, which we called penalized PSB. We gave a published formula and tried to improve it, but we couldn't solve some of the exhibited shortcomings of the formula.
We then developed a Secant Update version of the penalized PSB. This formula was surprisingly simpler.
Publications we found about pPSB are quite recent and we found no indication of a SU version of the formula. Here again, the track is at least less explored, and maybe new.

Finally, in Chapter 6, we have tested the different formulas on multiple standard problems in order to compare their performances.
The results for the generalized PSB are quite encouraging. As we have included the formula in a

simple algorithm. We should test it within more sophisticated algorithms, including line search, to see if the performance still increases.
The results for penalized PSB are not so good, but the weight definition has not been analyzed in detail. A deeper study is needed in order to discard the method or to find the best way to define the weights.

In conclusion, we have found some interesting new update formulas for the Quasi-Newton method. However, our work was mainly theoretical and there is still work to do to test the formulas. Next to the tests, there is some place for improvement and some open problems.

We finish our study by giving some way-ahead ideas to expand the present study:

- In general, the different formulas should be tested within a more complex algorithm (better filtering of the matrix S, better convergence criteria, use of `gmres` also for the inverse formulas, better choice of $B_0 \dots$). We should also potentially permit some step optimization of the step (for instance with line search when the steps start becoming too small).
- We should use the possibility given by Moré's procedure to test the formulas on flat function, on function with steep slopes and with starting point being farther away of the optimum point. This would give us a much better idea of the reliability and the robustness of the update formulas.
- The inversion of $S^T S$ in generalized PSB could lead to numerical problems and be a time consuming operation. Finding a way to avoid this inversion is maybe possible. The existence of Formula 3.8 and the work we have done in Section 5.2.2 are two examples of a similar situation where there is a way to avoid matrix inversion.
- The performance of penalized PSB should be analyzed in more details. The weights definition has probably an impact on the result. Different weights combinations should be tested in order to find the best one.
- The whole study could be adapted to positive definite matrices instead of symmetric matrices.
- A test on a real application will be a nice proof of the performances of the formulas.
- Theoretical proofs of convergence of the different formulas could be developed.
- The observed asymptotic rate of convergence (superlinear or not) of the different algorithms could also be defined.
- Instead of trying to satisfy every secant equations, we can also try to limit to a small number of the latest secants (2, 3, 5, 10 ...). This could be a good trade-off between the satisfying of the multisecant and symmetric properties on the one side, and a lower computation complexity and limited memory on the other side.

Bibliography

[1] Basic facts about Hilbert space. [Online; accessed 22-April-2017].

[2] Enrico Bertolazzi. Quasi-newton methods for minimization. http://www.ing.unitn.it/~bertolaz/2-teaching/2011-2012/AA-2011-2012-OPTIM/lezioni/slides-mQN.pdf, 2011.

[3] Peter Blomgren. Numerical optimization. http://terminus.sdsu.edu/SDSU/Math693a_f2009/Lectures/19/lecture.pdf, 2015.

[4] Stephen Boyd and Jon Dattorro. Alternating projections. *EE392o, Stanford University*, 2003.

[5] CG Broyden. On the discovery of the "good broyden" method. *Mathematical programming*, 87(2):209–213, 2000.

[6] Charles G Broyden. A class of methods for solving nonlinear simultaneous equations. *Mathematics of computation*, 19(92):577–593, 1965.

[7] Ward Cheney and Allen A Goldstein. Proximity maps for convex sets. *Proceedings of the American Mathematical Society*, 10(3):448–450, 1959.

[8] Haw-ren Fang and Yousef Saad. Two classes of multisecant methods for nonlinear acceleration. *Numerical Linear Algebra with Applications*, 16(3):197–221, 2009.

[9] S Gratton, V Malmedy, and Ph L Toint. Quasi-newton updates with weighted secant equations. *Optimization Methods and Software*, 30(4):748–755, 2015.

[10] Robby Haelterman. *Analytical study of the least squares quasi-Newton method for interaction problems*. PhD thesis, Ghent University, 2009.

[11] Anatoli Iouditski. Convex sets. [Online; accessed 22-April-2017].

[12] M Khan. Updating inverse of a matrix when a column is added. Technical report, Technical report, UBC, 2008. Unpublish, 2008.

[13] Klement and Jan. On using quasi-newton algorithms of the broyden class for model-to-test correlation. *Journal of Aerospace Technology and Management*, 6(4):407–414, 2014.

[14] Dominik Mielczarek. Minimal projections onto spaces of symmetric matrices. *Univ. Iagel. Acta Math*, 44:69–82, 2006.

[15] José Luis Morales. Variational quasi-newton formulas for systems of nonlinear equations and optimization problems. 2008.

[16] Jorge J Moré, Burton S Garbow, and Kenneth E Hillstrom. Testing unconstrained optimization software. *ACM Transactions on Mathematical Software (TOMS)*, 7(1):17–41, 1981.

[17] Michael JD Powell. A new algorithm for unconstrained optimization. *Nonlinear programming*, pages 31–65, 1970.

[18] MJD Powell. Beyond symmetric broyden for updating quadratic models in minimization without derivatives. *Mathematical Programming*, 138(1-2):475–500, 2013.

[19] Werner C Rheinboldt. Quasi-newton methods.

[20] Robert B Schnabel. Quasi-newton methods using multiple secant equations. Technical report, DTIC Document, 1983.

[21] Jack Sherman and Winifred J Morrison. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, 21(1):124–127, 1950.

[22] Paul Van Dooren. 2380 - matrix computations. 2015.

[23] Wikipedia. Computational complexity of mathematical operations. [Online; accessed 12-May-2017].

[24] Wikipedia. Sherman–morrison formula. [Online; accessed 12-April-2017].

[25] Wikipedia. Woodbury matrix identity. [Online; accessed 12-April-2017].

[26] Max A Woodbury. Inverting modified matrices. *Memorandum report*, 42:106, 1950.

[27] Chen YangQuan. Hybrid symbolic and numerical simulation studies of time-fractional order. [Online on The MathWorks, Inc. website; accessed 15-February-2017].

[28] Jianzhong Zhang and Chengxian Xu. Properties and numerical performance of quasi-newton methods with modified quasi-newton equations. *Journal of Computational and Applied Mathematics*, 137(2):269–278, 2001.

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve **www.uclouvain.be/epl**