M&C 2017 - International Conference on Mathematics & Computational Methods Applied to Nuclear Science & Engineering, Jeju, Korea, April 16-20, 2017, on USB (2017)

Self-Regulating Broyden Sequences For Solving Non-linear Equations In The GALILEO™ Fuel Rod Code

R. van Geemert*, C. Garnier**

*AREVA GmbH, Paul-Gossen-Strasse 100, Erlangen (Germany)
**AREVA NP, Rue Juliette Recamier 10, Lyon (France)
*rene.vangeemert@areva.com, **christophe.garnier@areva.com

Abstract - *For large-scale applications of the continuous thermal-mechanical model in AREVA NP's fuel rod code GALILEO™, embedded systems of non-linear equations must be solved many times over with high dependability and efficiency. This paper presents a compact overview of the non-linear equations to be solved, followed by a description of the developed Broyden solver setup that proved adequate for optimizing robustness and computational efficiency. This final setup includes self-regulating mechanisms such as adaptive restarts and adaptive underrelaxation. The purpose of this paper is to emphasize and demonstrate the importance of such embedded mechanisms in non-linear solver implementations. Verifications of the achieved solver properties, in terms of comparisons with simpler setups that lack adaptive measures, are presented in terms of associated observations on solver behavior for different test cases.*

I. INTRODUCTION

AREVA NP's fuel performance code GALILEO™ [1] accurately predicts the fuel rod thermal-mechanical behavior in nominal and off-nominal conditions for all AREVA NP rod designs [2] [1]. The GALILEO™ fuel rod model is based on a classical 1.5-dimensional approach similar to other modern fuel rod codes [3] [4]. As outlined in [5], the quasi-static mechanical problem is solved in terms of a finite element model for the fuel rod. This classical approach enables to take full methodology benefits from similar known finite element concepts [6] [7] [8] [9]. Particularly, the fuel cracking behavior is modeled through use of an anisotropic elastic damage model [10].

Sophisticated computational modeling of the fuel pellet's non-linear mechanical behavior can lead to non-trivial numerical challenges. This is mainly due to the requirement that the embedded solution processes must mutually consistently address the time evolution of several different coupled physical phenomena, such as viscoplasticity, stress-dependent swelling and cracking. As this must all be arranged in a computationally mutually consistent, multi-physics-coupled way, a well-designed non-linear coupled iterative solution process is needed for enabling this in an efficient and numerically stable manner.

Within this context, a specific need for a *quasi-Newton solver* arises when exact mutual consistence must be achieved between global equilibrium (formulated in terms of the finite element model) and local non-linear behavior of each element, whereas the determination of an explicit Jacobian (=multi-dimensional, multi-physics coupling gradient) for all coupled multi-physics system equations is either far too expensive computationally or infeasible for other reasons.

It is this particular need that is fulfilled in GALILEO™ by the specific Broyden solver implementation described in this article.

II. NON-LINEAR EQUATIONS TO BE SOLVED IN GALILEO™'S CONTINUOUS THERMO-MECHANICAL MODEL

One of the central equations of relevance in GALILEO™'s continuous thermal-mechanical model, to be solved several times over for each time step as part of the time integration process, is the *stiffness equation* [1]:

$$\hat{\mathbf{K}}\,\underline{u} \;=\; \underline{b} \tag{1}$$

Through the availability of the *stiffness matrix* $\hat{\mathbf{K}}$, this equation relates the array $\underline{u}$ of segment-wise displacements in fuel rod segments (per time step, as a function of burnup and due to influence of several thermo-mechanical processes) to a stress-determined source array $\underline{b}$. The typical $\hat{\mathbf{K}}$-matrix structure is displayed in Figs.1-3. For each time step, the non-linearity of this equation arises due to the property that, through implicit operations embedded in the time step integrator, the source array $\underline{b}$ co-depends on the solution, such that $\underline{b} = \underline{b}[\underline{u}]$, i.e. $\underline{b}$ is a multi-dimensional function of the displacement array $\underline{u}$.

Hence, when $\underline{u}$ is updated (and thereby perturbed with respect to the previous estimation) during the non-linear solution process, this unavoidably leads to a perturbation in the source term as well: $\underline{u} \rightarrow \underline{u}' = \underline{u} + \delta\underline{u} \implies \underline{b} \rightarrow \underline{b}' = \underline{b} + \delta\underline{b}$. Due to this, the non-linear solution process will include a dynamic interplay between solution updates $\underline{u}' = \underline{u} + \delta\underline{u}$ and source updates $\underline{b}' = \underline{b} + \delta\underline{b}$, until a mutually consistent convergence (characterized by successively smaller adjustments $\delta\underline{u}$ and $\delta\underline{b}$) towards the exact, asymptotic solution $\underline{u}_\infty$ and the exact source $\underline{b}[\underline{u}_\infty]$ is achieved. Formulated alternatively, the objective is thus to solve the non-linear equation

$$\hat{\mathbf{K}}\,(\underline{u} + \delta\underline{u}) \;=\; \underline{b}[\underline{u} + \delta\underline{u}] \tag{2}$$

When considering a localized linearization (around a local estimation for $\underline{u}$) of $\underline{b}[\underline{u} + \delta\underline{u}]$:

$$\underline{b}[\underline{u} + \delta\underline{u}] \cong \underline{b}[\underline{u}] + \frac{\partial \underline{b}}{\partial \underline{u}} \delta\underline{u} \tag{3}$$

it would obviously be extremely helpful to have the multi-dimensional gradient matrix $\frac{\partial \underline{b}}{\partial \underline{u}}$ available in an explicit, tractable form. Unfortunately, this is not the case. In the GALILEO^{TM} code, the perturbation $\delta\underline{b}[\delta\underline{u}]$, that results from the change $\delta\underline{u}$, follows as a result of an implicit chain of operations embedded in the time integration approach.

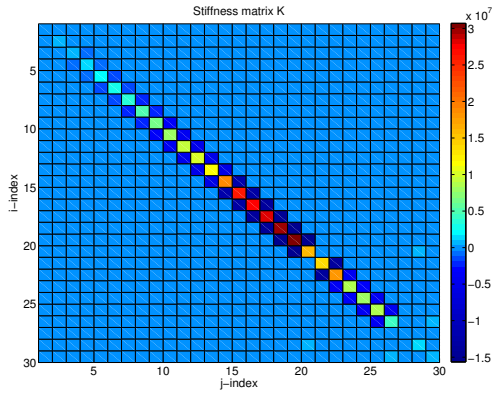

Fig.1. Structure of element values in a sample stiffness matrix $\hat{\mathbf{K}}$.

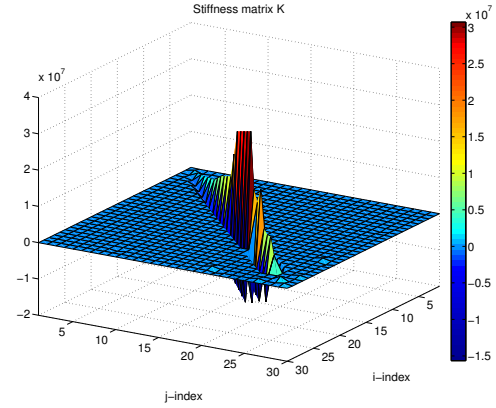

Fig.2. Surface plot of structure of element values in a sample stiffness matrix $\hat{\mathbf{K}}$.

It is especially for solving such systems, that include also implicit operators, that the *Broyden approach* [11] [12] as non-linear solution algorithm presents practical advantages. The Broyden approach does not require the explicit availability of multi-dimensional gradient matrices: it requires only the black box capability of computing the perturbation $\delta\underline{b}[\delta\underline{u}]$ that results from a change $\delta\underline{u}$.

In that sense, the Broyden approach is the multi-dimensional equivalent of the more widely known *secant approach* [12] for zero-dimensional non-linear equations. Between Broyden steps, for each successively updated source vector $\underline{b}'$, the updated displacement array solution $\underline{u}'$ is solved directly from

$$\hat{\mathbf{K}}\, \underline{u}' = \underline{b}' \tag{4}$$

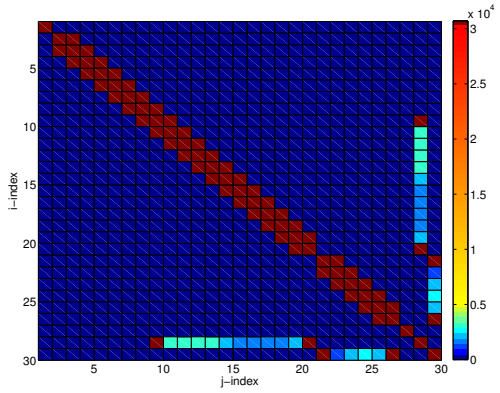

Fig.3. Logarithmic zoom on absolute values in off-diagonal structure outside of the tridiagonal band (latter is larger than 10^5) of the sample stiffness matrix $\hat{\mathbf{K}}$.

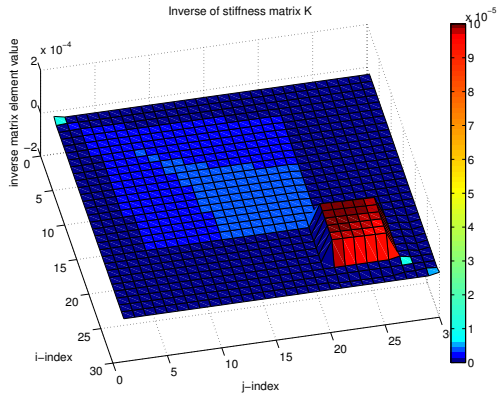

Fig.4. Surface plot of the inverse matrix $\hat{\mathbf{K}}^{-1}$.

such that $\underline{u}'$ can be written symbolically as:

$$\underline{u}' = \hat{\mathbf{K}}^{-1}\, \underline{b}' \tag{5}$$

Figs.1-4 present graphical illustrations of what the stiffness matrix $\hat{\mathbf{K}}$ and its inverse $\hat{\mathbf{K}}^{-1}$ typically look like (with the explicit $\hat{\mathbf{K}}^{-1}$ having been obtained through use of a separate routine based on the use of BiCGStab). For $\hat{\mathbf{K}}$ itself, it is visible clearly that it is extremely sparse in terms of most of its matrix elements being equal to zero.

Within the context of solving the stiffness equations $\hat{\mathbf{K}}\,\underline{u}' = \underline{b}'$ (Eq.(1)) many times over (and obviously also with varying $\hat{\mathbf{K}}$), GALILEO™ does not explicitly compute and store the inverse $\hat{\mathbf{K}}^{-1}$ of the stiffness matrices $\hat{\mathbf{K}}$. Instead, an *LU-decomposition* [12] is formed for each new $\hat{\mathbf{K}}$, such that $\hat{\mathbf{K}} = \hat{\mathbf{L}}\hat{\mathbf{U}}$, with $\hat{\mathbf{L}}$ a lower-diagonal matrix and $\hat{\mathbf{U}}$ an upper-diagonal matrix. This LU-decomposition needs to be newly determined once per time integration step. After that, it remains constant throughout the entire subsequent non-linear solution process for that individual time step.

Once $\hat{\mathbf{L}}$ and $\hat{\mathbf{U}}$ are available, the displacement array $\underline{u}$ can be solved from $\hat{\mathbf{K}}\,\underline{u} = \hat{\mathbf{L}}\hat{\mathbf{U}}\,\underline{u} = \underline{b}$ in two steps. First, the intermediate array $\underline{y}$ is solved from:

$$\hat{\mathbf{L}}\,\underline{y} = \underline{b} \tag{6}$$

followed by solution of the displacement array $\underline{u}$ from:

$$\hat{\mathbf{U}}\,\underline{u} = \underline{y} \tag{7}$$

These steps boil down to relatively inexpensive, direct Gaussian elimination procedures. Within this context, it is interesting to note that $\hat{\mathbf{L}}$ and $\hat{\mathbf{U}}$ are just as sparse as the stiffness matrix $\hat{\mathbf{K}}$ itself. Due to this, most multiplications per matrix-vector operation are actually non-contributing multiplications with zero.

It is noted as well that, following its determination at each individual time integration step, an available LU-decomposition is applied several times in succession during the Broyden solution process. This clearly constituted a potential for measurable reduction of the associated computational effort. Hence, as an additional code modification, a dedicated compressed LU-decomposition representation was programmed that *avoided* premultiplication with the many zeroes that are typically present in the LU-decomposed form of the stiffness-matrix.

Additionally, it was identified that the previous setup of the computational process of establishing the LU-decomposition could be optimized for the sparse matrix context in GALILEO™. The first version of our LU-routine, that was obtained through download from a public web library, treated the inputted matrices as *dense* in spite of those being *extremely sparse*. Due to this, by far most of the process-internal multiplications (in the computation of process-relevant vector outer products) were, again, multiplications among zero-valued matrix elements. In a fully dense treatment, the process of determining the LU-decomposition comprises $O(n^3/3)$ multiplicative operations. Thus, it paid off to create an application-tailored, 'sparse' LU-decomposition routine version that, in a structured way, automatically excludes *a priori* the many multiplications with zeroes.

The combinations of these measures enabled lean, computationally cheap operations for solving the successive stiffness equations.

III. NONLINEARITY BY CO-DEPENDENCE OF SOURCE UPON SOLUTION

One needs to realize that, during the non-linear iterative solution process, the source $\underline{b}'$ is associated with a previous estimation $\underline{u}$ for the displacement array. Writing $\underline{u}' = \underline{u} + \delta\underline{u}$, the ideal choice for $\delta\underline{u}$ would be the one that anticipates its future influence on the source vector $\underline{b}'$. Under the influence of $\delta\underline{u}$, the latter will change to a further perturbed source term $\underline{b}''$ with

$$\underline{b}'' \cong \underline{b}' + \frac{\partial \underline{b}}{\partial \underline{u}}\,\delta\underline{u} \tag{8}$$

Hence the ideal choice for $\delta\underline{u}$ would be the one that fulfils

$$\left(\hat{\mathbf{I}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial \underline{b}}{\partial \underline{u}}\right)\delta\underline{u} + \underline{r} = \underline{0} \tag{9}$$

with $\underline{r}$ being, at that point of the iterative solution process, the residual of the overall (non-linear) equation as associated with the previous updated estimation $\underline{u}$ and the source vector $\underline{b}[\underline{u}]$ associated with $\underline{u}$:

$$\underline{r} = \underline{u} - \hat{\mathbf{K}}^{-1}\,\underline{b}[\underline{u}] \tag{10}$$

If the operator $\frac{\partial \underline{b}}{\partial \underline{u}}$ would be known and be given explicitly, Eq.(9) would boil down to a *Newton equation* [12] for the successive updates $\delta\underline{u}$:

$$\hat{\mathbf{J}}\,\delta\underline{u} + \underline{r} = \underline{0} \tag{11}$$

with the Jacobian being:

$$\hat{\mathbf{J}} = \hat{\mathbf{I}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial \underline{b}}{\partial \underline{u}} \tag{12}$$

such that the ideal $\delta\underline{u}$ (whether solved exactly or approximately as improved update from Eq.(9)) can be written symbolically as:

$$\delta\underline{u} \;=\; -\;\hat{\mathbf{J}}^{-1}\,\underline{r} \tag{13}$$

Successive updates for the displacement array would hence be defined as:

$$\begin{aligned}
\underline{u}^{(n+1)} &= \underline{u}^{(n)} + \delta\underline{u}^{(n)} \\
&= \underline{u}^{(n)} - \hat{\mathbf{J}}^{-1}\,\underline{r}^{(n)} \\
&= \underline{u}^{(n)} - \hat{\mathbf{J}}^{-1}\left(\underline{u}^{(n)} - \hat{\mathbf{K}}^{-1}\,\underline{b}[\underline{u}^{(n)}]\right) \\
&= \underline{u}^{(n)} - \left(\hat{\mathbf{1}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial\underline{b}}{\partial\underline{u}}\right)^{-1}\,\underline{r}^{(n)}
\end{aligned} \tag{14}$$

However, in GALILEO^{TM} the operator $\frac{\partial\underline{b}}{\partial\underline{u}}$ is not available in explicit form, due to which either $\tilde{\mathbf{J}} = \hat{\mathbf{1}}$ is to be used as a (very) poor approximation to the Jacobian, or a more intelligent algorithm (such as Broyden's) is applied that somehow does manage to establish a better approximation of $\hat{\mathbf{J}}$ in terms of being somewhere between $\tilde{\mathbf{J}} = \hat{\mathbf{1}}$ and the unknown exact Jacobian $\hat{\mathbf{J}} = \hat{\mathbf{1}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial\underline{b}}{\partial\underline{u}}$.

IV. CONVERGENCE ANALYSIS

For providing a theoretical framework for convergence analysis, we now legitimately assume the existence of an exact, asymptotic, fully converged solution $\underline{u}_\infty$, and rewrite $\underline{u}^{(n)}$ as variation $\underline{u}_\infty + \delta\underline{u}^{(n)}$ around $\underline{u}_\infty$. Obviously, upon full convergence of the non-linear iterative solution process, we obtain $\delta\underline{u}^{(n)} \to \underline{0}$ and $\underline{u}^{(n)} \to \underline{u}_\infty$ for some sufficiently large value of n. We can now rewrite Eq.(14) as:

$$\begin{aligned}
\underline{u}_\infty + \delta\underline{u}^{(n+1)} &= \underline{u}_\infty + \delta\underline{u}^{(n)} \\
&- \hat{\mathbf{J}}^{-1}\left(\underline{u}_\infty + \delta\underline{u}^{(n)} - \hat{\mathbf{K}}^{-1}\,\underline{b}[\underline{u}_\infty + \delta\underline{u}^{(n)}]\right) \\
&\cong \underline{u}_\infty + \delta\underline{u}^{(n)} \\
&- \hat{\mathbf{J}}^{-1}\left(\underline{u}_\infty + \delta\underline{u}^{(n)} - \hat{\mathbf{K}}^{-1}\left(\underline{b}[\underline{u}_\infty] + \frac{\partial\underline{b}}{\partial\underline{u}}\right)\delta\underline{u}^{(n)}\right)
\end{aligned} \tag{15}$$

Since obviously the exact, fully converged solution $\underline{u}_\infty$ will exactly fulfil the equality $\underline{u}_\infty = \hat{\mathbf{K}}^{-1}\underline{b}[\underline{u}_\infty]$, Eq.(15) boils down to the following equation for relating the new solution adjustment $\delta\underline{u}^{(n+1)}$ to the previous solution adjustment $\delta\underline{u}^{(n)}$:

$$\begin{aligned}
\delta\underline{u}^{(n+1)} &= \delta\underline{u}^{(n)} - \hat{\mathbf{J}}^{-1}\left(\delta\underline{u}^{(n)} - \hat{\mathbf{K}}^{-1}\,\frac{\partial\underline{b}}{\partial\underline{u}}\,\delta\underline{u}^{(n)}\right) \\
&= \left(\hat{\mathbf{1}} - \hat{\mathbf{J}}^{-1}\left(\hat{\mathbf{1}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial\underline{b}}{\partial\underline{u}}\right)\right)\delta\underline{u}^{(n)}
\end{aligned} \tag{16}$$

With the error decay operator $\hat{\mathbf{\Theta}}$ defined as:

$$\hat{\mathbf{\Theta}} \;=\; \hat{\mathbf{1}} - \hat{\mathbf{J}}^{-1}\left(\hat{\mathbf{1}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial\underline{b}}{\partial\underline{u}}\right) \tag{17}$$

we obtain the simply error decay expression:

$$\delta\underline{u}^{(n+1)} \;=\; \hat{\mathbf{\Theta}}\,\delta\underline{u}^{(n)} \tag{18}$$

If the operator $\hat{\mathbf{\Theta}}$ generally imposes sign changes per individual array element $\delta\underline{u}_i^{(n)}$, the error decay will be of an oscillatory nature regardless of whether convergence is progressing or stagnating. That this is indeed the case has been confirmed for numerical observations concerning both well-converging and not-so-well-converging scenarios. Obviously, convergence is enabled only if the operator norm $\| \hat{\mathbf{\Theta}} \|$ is smaller than 1.

Clearly, if the Jacobian would exactly fulfil the desired property $\hat{\mathbf{J}}^{-1} = \left(\hat{\mathbf{1}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial\underline{b}}{\partial\underline{u}}\right)^{-1}$, then $\hat{\mathbf{\Theta}}$ would be the null operator $\hat{\mathbf{0}}$ and the iterative solution process according to Eq.(14) would converge practically immediately (typically not fully immediately due to some remaining inaccuracy in the linearization $\underline{b}[\underline{u}_\infty + \delta\underline{u}^{(n)}] \cong \underline{b}[\underline{u}_\infty] + \frac{\partial\underline{b}}{\partial\underline{u}}\,\delta\underline{u}^{(n)}$, but very fast nonetheless).

As pointed out in the previous section, the operator $\frac{\partial\underline{b}}{\partial\underline{u}}$ is not available in explicit form, due to which either $\tilde{\mathbf{J}} = \hat{\mathbf{1}}$ is to be used as poor approximation to the Jacobian, or a more intelligent algorithm (such as Broyden's) is to be applied that somehow does manage to establish a better approximation of $\hat{\mathbf{J}}$ in terms of being somewhere between $\tilde{\mathbf{J}} = \hat{\mathbf{1}}$ and the unknown exact Jacobian $\hat{\mathbf{J}} = \hat{\mathbf{1}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial\underline{b}}{\partial\underline{u}}$.

This means that the convergence efficiency of the Broyden algorithm is positioned somewhere between the (usually poor) convergence efficiency of a simple Jacobi approach (i.e. with $\tilde{\mathbf{J}} = \hat{\mathbf{1}}$) and the theoretically ideal immediate convergence as enabled by the theoretically ideal property $\hat{\mathbf{J}} = \hat{\mathbf{1}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial\underline{b}}{\partial\underline{u}}$. This efficiency positioning has, in any case, been confirmed for the self-regulated, adaptively restarted variant of the Broyden algorithm that is presented in the next section.

V. BROYDEN APPROACH FOR HANDLING INVISIBLE NONLINEARITY IN THE ITERATIVE SOLUTION PROCESS

The Broyden algorithm [11] [12] starts with the best practical initial guess for the Jacobian, which here is $\tilde{\mathbf{J}} = \hat{\mathbf{1}}$. Given an initial guess $\underline{u}_{init}$ for the displacement array $\underline{u}$, $\delta\underline{u}_{init}$ then follows from:

$$\delta\underline{u}_{init} \;=\; -\;\underline{r}_{init} \;=\; \hat{\mathbf{K}}^{-1}\,\underline{b}(\underline{u}_{init}) \;-\; \underline{u}_{init} \tag{19}$$

after which the displacement array is updated as $\underline{u}' = \underline{u} + \delta\underline{u}$, from which the updated source can be computed as $\underline{b}' = \underline{b}[\underline{u}']$. Subsequently, a new equation residual $\underline{r}' = \hat{\mathbf{K}}^{-1}\,\underline{b}'[\underline{u}'] - \underline{u}'$ can be computed. According to Broyden's approach, a meaningful update of the approximated Jacobian can now follow from the possibility of determining a $\delta\hat{\mathbf{J}}$ that fulfils the secant condition

$$\left(\hat{\mathbf{J}} + \delta\hat{\mathbf{J}}\right)\,\delta\underline{u} \;=\; \delta\underline{r} \;=\; \underline{r}' - \underline{r} \;=\; \underline{r}'[\underline{u}+\delta\underline{u}] - \underline{r}[\underline{u}] \tag{20}$$

Simultaneously, one also needs to fulfil the property that this Jacobian update will operate exclusively along the direction of $\delta\underline{u}$, due to this being the direction along which this part of the multi-gradient information could be sampled. This latter property is fulfilled if

$$\delta\hat{\mathbf{J}}\,\underline{q} \;=\; 0 \;\;\forall\,\underline{q} \;\perp\; \underline{u} \;:\; \underline{q}^{T}\delta\underline{u} \;=\; 0 \tag{21}$$

The Broyden solution that fulfils both Eqs.(20),(21) is the following outer product:

$$\delta\hat{\mathbf{J}} \;=\; \left(\delta\underline{r} \;-\; \hat{\mathbf{J}}\,\delta\underline{u}\right)\,\frac{\delta\underline{u}^{T}}{\delta\underline{u}^{T}\,\delta\underline{u}} \tag{22}$$

such that the successive approximations for the Jacobian, along previously covered update directions during the solution process, follow from:

$$\hat{\mathbf{J}}' \;=\; \hat{\mathbf{J}} \;+\; \left(\delta\underline{r} \;-\; \hat{\mathbf{J}}\,\delta\underline{u}\right)\,\frac{\delta\underline{u}^{T}}{\delta\underline{u}^{T}\,\delta\underline{u}} \tag{23}$$

Providing indices n for the successive updates, we obtain:

$$\hat{\mathbf{J}}_{n+1} \;=\; \hat{\mathbf{J}}_{n} \;+\; \left(\delta\underline{r}_{n} \;-\; \hat{\mathbf{J}}_{n}\,\delta\underline{u}_{n}\right)\,\frac{\delta\underline{u}_{n}^{T}}{\delta\underline{u}_{n}^{T}\,\delta\underline{u}_{n}} \tag{24}$$

In order to avoid the (iterative) solution effort for solving (11), it is possible to apply the *Sherman-Morrison* [Burden 2011] formula for direct determination and successive updates of the *inverse* of the approximated Jacobian:

$$\hat{\mathbf{J}}_{n+1}^{-1} \;=\; \hat{\mathbf{J}}_{n}^{-1} \;+\; \left(\delta\underline{u}_{n} \;-\; \hat{\mathbf{J}}_{n}^{-1}\,\delta\underline{r}_{n}\right)\,\frac{\delta\underline{u}_{n}^{T}\,\hat{\mathbf{J}}_{n}^{-1}}{\delta\underline{u}_{n}^{T}\,\hat{\mathbf{J}}_{n}^{-1}\,\delta\underline{r}_{n}} \tag{25}$$

This approach enables the solution of the successive $\underline{u}_n$ through combined use of Eqs.(13),(25). In practical implementations, it may be convenient to utilize the property

$$\delta\underline{u}_{n}^{T}\,\hat{\mathbf{J}}_{n}^{-1} \;=\; \left(\left(\hat{\mathbf{J}}_{n}^{-1}\right)^{T}\,\delta\underline{u}_{n}\right)^{T} \tag{26}$$

with $\left(\hat{\mathbf{J}}_{n}^{-1}\right)^{T}$ the transpose of $\hat{\mathbf{J}}_{n}^{-1}$, such that Eq.(25) can be rearranged as:

$$\hat{\mathbf{J}}_{n+1}^{-1} \;=\; \hat{\mathbf{J}}_{n}^{-1} \;+\; \left(\delta\underline{u}_{n} \;-\; \hat{\mathbf{J}}_{n}^{-1}\,\delta\underline{r}_{n}\right)\,\frac{\left(\left(\hat{\mathbf{J}}_{n}^{-1}\right)^{T}\,\delta\underline{u}_{n}\right)^{T}}{\left(\left(\hat{\mathbf{J}}_{n}^{-1}\right)^{T}\,\delta\underline{u}_{n}\right)^{T}\,\delta\underline{r}_{n}} \tag{27}$$

This is how the updates of the inverse Jacobian were programmed. The update process

$$\underline{u}^{(n+1)} \;=\; \underline{u}^{(n)} \;-\; \hat{\mathbf{J}}_{n+1}^{-1}\,\underline{r}^{(n)} \tag{28}$$

can now be expected to enable a convergence efficiency that is positioned somewhere between the convergence efficiency of a simple Jacobi approach (i.e. with $\tilde{\mathbf{J}} = \hat{\mathbf{1}}$) and the theoretically ideal immediate convergence as enabled by the theoretically ideal property $\hat{\mathbf{J}}^{-1} = \left(\hat{\mathbf{1}} - \hat{\mathbf{K}}^{-1}\,\frac{\partial\underline{b}}{\partial\underline{u}}\right)^{-1}$.

In order to monitor convergence of the overall solution process, a single estimator was established for the achieved relative solution accuracy that is unbiased by the overall magnitude level of the solution. This estimator additionally enables a well-balanced, neutral convergence criterion ϵ_{n+1} to be used for deciding whether the solution process can be truncated or not:

$$\epsilon_{n+1} \;=\; \max_{i}\left\{\frac{|u_{i}^{(n+1)} - u_{i}^{(n)}|}{\frac{1}{N}\sum_{i}^{N}|u_{i}^{(n)}|}\right\} \tag{29}$$

Here, N denotes the length of the solution array (with the matrix $\hat{\mathbf{K}}$ being of dimension N x N) and i denotes the array index for the iterand $\underline{u}$.

VI. SELF-REGULATING BROYDEN SEQUENCES: ADAPTIVE SEQUENCE TRUNCATIONS, RESTARTS AND UNDERRELAXATIONS

During development and testing of the Broyden solver, it quickly proved necessary to include a *restart capability*. This was especially true in case the Broyden process had not yet provided a converged result after, say, 10 steps. Obviously, this also depends on the numerical distance between the initial solution guess and the final solution.

Since the Broyden process basically builds local, linearized multi-dimensional gradient information, it makes sense to repeat this around a successively improved initial guess (as delivered by an earlier pursued Broyden sequence) that gets closer and closer to the final solution. As argued in [13], such restarts may then lead to globally improved convergence behavior and, in this way, enable run time savings as well. Similar experiences have been made for other kinds of applications in other Broyden implementation studies [14] [15] [16].

Building further upon this idea, in constructing the new Broyden solver for the thermo-mechanical loop in GALILEO^{TM}, a *self-regulating, adaptive restart mechanism* was programmed. This includes adaptive thresholds t_{trunc} for deciding when, based on tracking the residual decay from the start of the Broyden sequence, the residual has decayed sufficiently, such that a sequence restart makes sense.

This means that, at the start of each Broyden sequence, one computes and stores the value of the initial maximum relative residual $|r_{init}|$ associated with the initial solution $\underline{u}_{init}$ of that sequence. During the sequence, one then systematically computes the maximum relative residuals $|r_n|$ associated with the next, Broyden-computed solution iterands $\underline{u}_n$. The criterion for truncating the current Broyden sequence, and starting a new one, is:

$$|r_n| < t_{trunc}\,|r_{init}| \implies restart \tag{30}$$

The truncation parameter t_{trunc} was set to be 0.1 initially, and is increased gradually (up to 0.3, for enforcing shorter sequences) if no convergence has been established after 10 Broyden sequences. Other adaptive measures were implemented, such as the following one that triggers a restart if, after 3 Broyden steps, it is observed that the residual goes up instead of down:

$$n > 3 \text{ and } |r_{n+1}| > |r_n| \implies restart \tag{31}$$

Furthermore, maximum sequence lengths were defined according to how many restarted sequences have been pursued already, and initially the following adaptive underrelaxations of solution updates were applied:

$$\underline{u}_{n+1} = \underline{u}_n + \alpha\,\delta\underline{u}_{n+1} \tag{32}$$

in which the underrelaxation factors α were lowered gradually and mildly as the number of already pursued Broyden sequences increased. Actually, later developments in response to some yet remaining convergence issues led to a modified adaptive underrelaxation approach that resulted in a full replacement of the approach expressed in Eq.(32). This is discussed later on in this paper.

VII. VERIFICATIONS

Verifications of the achieved improvements were provided in terms of observed solver property differences for several test cases. For all cases, and in an absolutely similar manner, the following improvements were easily observable:

- By comparison with unlimited Broyden sequences (no restarts, as featured in the previous Broyden solver), it was confirmed for all cases that substantially fewer time steps feature non-convergence problems (cf. Fig.5)
- By comparison with unlimited Broyden sequences, it was confirmed for all cases that most of the iterations converged two up to three times as fast due to the use of self-regulating, adaptive Broyden sequences (cf. Fig.6)

Nonetheless, in spite of the availability of the improved Broyden solver, all verification computations had also shown remaining issues of convergence stagnation with varying degrees of severity.

This becomes visible in Figs. 5 and 6 in terms of the remaining upward spikes. Here, the Broyden solver was unable to achieve a final convergence accuracy comparable to what was achievable for by far most of the other time steps.

In Fig.6, such occurrences are visible for the version featuring unlimited Broyden sequences (no restarts) by the truncation limit of 300 Broyden steps having been reached, without convergence having been achieved for those cases. These occurrences are not visible in the same manner for the new Broyden solver, since the new Broyden solver avoids unlimited continuations of stagnating Broyden sequences.

Although the achieved levels of relative convergence accuracy did seem reasonably adequate for allowing continuation of the computation, the unpredictability of these convergence stagnation occurrences remained a source of robustness concern. Due to this, this problem was investigated and, fortunately, it could be resolved by a modified underrelaxation setup that actually proved rather easy to implement.

VIII. ANALYSIS AND RESOLUTION OF OCCASIONAL CONVERGENCE STAGNATION SCENARIOS

During pursued in-depth analysis studies of the previously observed convergence stagnation problems, it became clear that the Broyden update process entered *non-converging and indefinitely self-repeating limit cycles*. The latter were characterized by rather substantial non-decaying oscillations in updates on the Broyden-approximated inverse of the Jacobian.

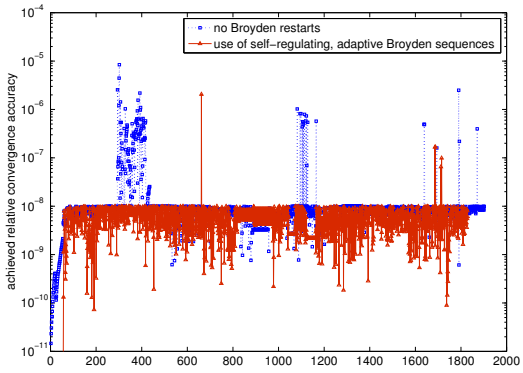

Fig.5. Final relative convergence accuracies as achievable when including self-regulation, adaptive restarts in the Broyden solver setup (compared to the unregulated Broyden setup).

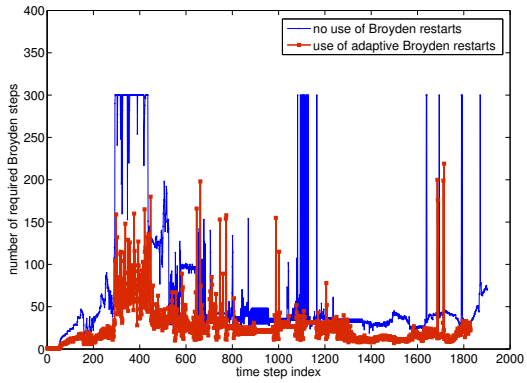

Fig.6. Numbers of iteration steps as needed when including self-regulation, adaptive restarts in the Broyden solver setup (compared to the unregulated Broyden setup).

As is typical for secant approaches, this kind of behavior usually introduced itself as soon as a reasonable accuracy of the solution had been achieved already, that however was definitely not yet commensurate with the tolerance-imposed required level of final convergence accuracy. The additional occurrences of sudden very small values of the denominator in the multi-dimensional gradient inverse update, that are not uncommon in any progressed secant procedure, then gave rise to drastically amplified manifestations of unintended deviations in the Broyden-approximated inverse of the Jacobian. The latter then affected solution updates in a significant and clearly unintended way, such that rather quickly the Broyden update process entered a unintended, non-converging, indefinitely self-repeating limit cycle that prevented the required full convergence of the solution.

It turned out to be possible to cure these convergence stagnations by imposing a new, modified underrelaxation setup. Due to knowing the potential merit of adaptive underrelaxation in other areas, a simple damping measure was in fact programmed already (cf. Eq.(32)) as part of the earlier improvement package for GALILEO^{TM}'s Broyden solver. Unfortunately, this particular damping implementation did not enable a resolution of the previously remaining convergence stagnation problems.

The basic insight on why the previous adaptive underrelaxation measure, as set in a multi-dimensional Broyden solution process, did not resolve the problem is the following: it merely moderated, along exactly the same multi-dimensional direction of solution adjustment, the magnitudes of the successive solution adjustments, while afterwards *not* influencing (in terms of a stabilizing damping effect) the update in the Broyden-approximated inverse of the Jacobian. Hence, the above-mentioned mechanism of repeated non-decaying oscillatory disturbances in the Jacobian inverse approximation was not changed by this particular underrelaxation measure. And thus the convergence stagnation problems remained, in spite of the underrelaxation measure according to Eq.(32) being activated.

In order to resolve this, a new and different adaptive underrelaxation setup, that the above-mentioned recent insight led to, was realized. Upon its activation (as auto-triggered in case of detected convergence stagnation), this enabled the needed damping effect on both Jacobian inverse approximation and (thereby also) the solution updates.

As will become clear from the derivation pursued in the next section, the decisive difference with the previously applied adaptive underrelaxation approach is that activated damping operations are now channeled through the Jacobian inverse updates, thereby directly calming down oscillations in the multi-dimensional gradient updates $\hat{\mathbf{J}}_n^{-1}$. As an indirect beneficial result of that, this then also leads to damping of the oscillatory updates in the solution iterands $\underline{u}^{(n)}$. In this way, the convergence behavior is typically still of an oscillatory nature, but the damping measure nonetheless enforces the necessary magnitude decay of the error in the successive Jacobian approximations $\hat{\mathbf{J}}_n^{-1}$ and in the solution iterands $\underline{u}^{(n)}$. Consequently, for all investigated cases, this approach proved effective as resolution measure for getting rid of all previously observed convergence stagnations.

IX. A NEW ADAPTIVE UNDERRELAXATION SETUP AT RESIDUAL FUNCTION EVALUATIONAL LEVEL

This resolution measure was derived from a rearrangement of the non-linear stiffness equations that eventually turned out to enable the needed preconditioning effect. Based on recent related investigations, the latter can be *bent* in a parametrized way (while nonetheless remaining fully equivalent to the original equations), such that its eigenspectral properties can become substantially more agreeable.

The associated new damping parameter α can be determined in a fully automated, adaptive way. From a certain perspective, it thereby acts as a *self-regulating lever* for automatically enforcing numerical stability of the non-linear solution process. In the new setup, we imposed the following rearrangement of Eq.(5), with $0 < \alpha \leq 1$, and emphasize that an exact solution of this rearranged equation is also an exact solution of Eq.(5):

$$\underline{u}' \;=\; \alpha\, \hat{\mathbf{K}}^{-1}\, \underline{b}'[\underline{u}] \;+\; (1-\alpha)\, \underline{u} \tag{33}$$

Hence it would be legitimate to pursue the non-linear solution process based on Eq.(33) instead of based on Eq.(5), if this would enable advantages for computational stability or computational efficiency (or both). Compared with Eq.(10), we then obtain the following expression for the residual to be applied in determining the successive Broyden updates for the inverse Jacobian:

$$\underline{r}_{\alpha} \;=\; \underline{u}_{\alpha}^{'} \;-\; \underline{u} \;=\; \alpha \Big(\hat{\mathbf{K}}^{-1}\, \underline{b}'[\underline{u}] \;-\; \underline{u} \Big) \;=\; \alpha\, \underline{r} \tag{34}$$

after which the displacement array is updated as $\underline{u}' = \underline{u} + \delta\underline{u}$, from which the newly updated source can be computed as $\underline{b}'' = \underline{b}'[\underline{u}']$. Subsequently, the new equation residual $\underline{r}_{\alpha}^{'} = \alpha\hat{\mathbf{K}}^{-1}\underline{b}'(\underline{u}') - \alpha\underline{u}'$ can be computed. According to Broyden's approach, a meaningful (and accordingly damped) update of the approximated Jacobian can now follow from the possibility of determining a $\delta\hat{\mathbf{J}}_{\alpha}$ that fulfils the secant condition

$$\Big(\hat{\mathbf{J}}_{\alpha} + \delta\hat{\mathbf{J}}_{\alpha}\Big)\,\delta\underline{u} = \delta\underline{r}_{\alpha} = \underline{r}_{\alpha}^{'} - \underline{r}_{\alpha} = \underline{r}_{\alpha}^{'}[\underline{u}+\delta\underline{u}] - \underline{r}_{\alpha}[\underline{u}] = \alpha\,\underline{r} \tag{35}$$

while also fulfilling the property that this Jacobian update will operate exclusively along the direction of $\delta\underline{u}$, due to this being the direction along which this part of the multi-gradient information could be sampled. Following exactly the same derivations as pursued earlier in this paper for the Broyden process, we arrive at:

$$\hat{\mathbf{J}}_{\alpha\,;\,n+1}^{-1} \;=\; \hat{\mathbf{J}}_{\alpha\,;\,n}^{-1} + \frac{1}{\alpha}\Big(\delta\underline{u}_{n} \;-\; \alpha\,\hat{\mathbf{J}}_{\alpha\,;\,n}^{-1}\,\delta\underline{r}_{n}\Big)\, \frac{\Big(\Big(\hat{\mathbf{J}}_{\alpha\,;\,n}^{-1}\Big)^{T}\,\delta\underline{u}_{n}\Big)^{T}}{\Big(\Big(\hat{\mathbf{J}}_{\alpha\,;\,n}^{-1}\Big)^{T}\,\delta\underline{u}_{n}\Big)^{T}\,\delta\underline{r}_{n}} \tag{36}$$

This is how the updates of the inverse Jacobian were programmed. The update process

$$\underline{u}^{(n+1)} \;=\; \underline{u}^{(n)} \;-\; \hat{\mathbf{J}}_{\alpha\,;\,n+1}^{-1}\,\underline{r}_{\alpha}^{(n)} \;=\; \underline{u}^{(n)} \;-\; \alpha\,\hat{\mathbf{J}}_{\alpha\,;\,n+1}^{-1}\,\underline{r}^{(n)} \tag{37}$$

can now be expected to enable, in a damped manner (if $0 < \alpha < 1$), a convergence behavior that is thus auto-steered to be a good trade-off between run time minimization and computational stability. If no convergence problems are detected, then obviously $\alpha = 1$ per default (and initial) value setting, and then this modified Broyden scheme remains unchanged compared to the original one. However, if convergence problems do occur and are detected accordingly, then α can be lowered gradually as a function of the number of pursued Broyden steps (i.e. $0 < \alpha < 1$ with gradually lowered values for α), until the overall solution process has come to full convergence.

This modified underrelaxation approach has enabled the resolution of all previously remaining convergence problems. It is emphasized that, per pursued Broyden sequence (i.e. between adaptively triggered restarts) the value for α needs to remain fixed, since the accumulated multi-dimensional gradient information must pertain to a fixed α-dependent expression for the equation residual. Hence the value for α can be changed / lowered only upon initiating a new, restarted Broyden sequence.

X. MORE VERIFICATIONS

Verifications of the achieved resolution of previously remaining convergence stagnations have been pursued for some known test cases. As can be seen in Fig.7, these remaining nonconvergences disappeared.

It must be admitted and emphasized, however, that, for these (fortunately few) time steps for which the undamped restarted Broyden sequences did not converge, a solution converged up to 10^{-8} is obtained at a rather high computational price to be paid for that. This is illustrated in Figs.8 and 9 that show comparisons (for the first and second occurring convergence stagnations in the concerned test case) between error decay curves when activating the modified damping measure (slow error decay but converged solution obtained) vs. when not using this damping measure (stagnating error decay and no convergence).

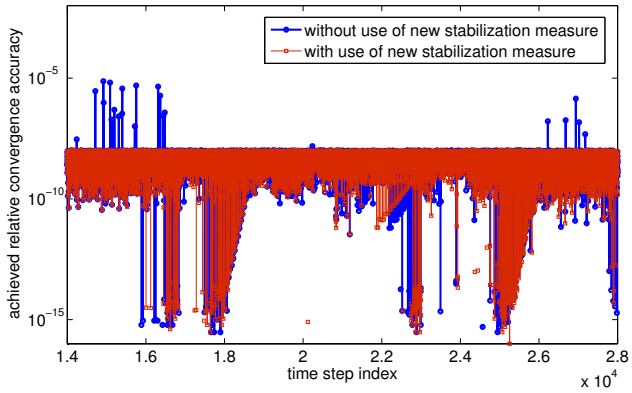

Fig.7. Illustration of resolution of previously remaining convergence stagnations.

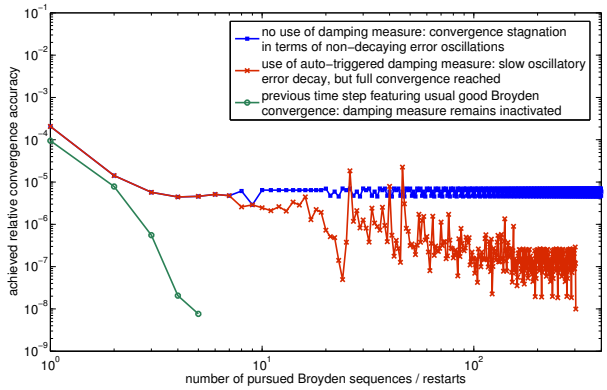

Fig.8. Illustration of convergence curves under different conditions, 1st non-convergence in the time integration sequence.

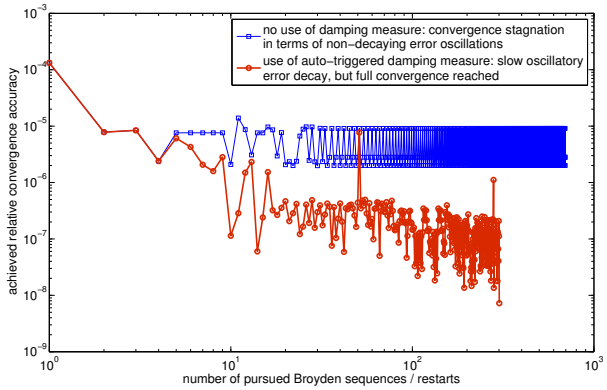

Fig.9. Illustration of convergence curves under different conditions, 2nd non-convergence in the time integration sequence.

As one can see in Figs.8 and 9, a solution converged up to 10^{-7} is obtained much earlier and hence at much lower computational cost. The upward spikes in the red curves in Fig.7 mark triggered modifications in the damping factor α. In Fig.8, the fully uncomplicated Broyden convergence curve (with no damping needed at all) for the time step prior to the first "problem time step" is shown.

It is emphasized that, per such run, perhaps between 10 and 20 such time steps were included (out of typically several hundreds up to several thousands of time steps) for which convergence stagnations manifest themselves, and for which hence the modified adaptive damping measure according to Eq.(33) is auto-triggered and deployed. Due to this, the overall computational price attached to the presently implemented stabilization measure is eventually modest and acceptable.

Possibly, the pursuit of more methodology R&D could enable a further improved resolution approach for computationally challenging time steps, that also give rise to lower additional computational burdens compared to what is typically happening when adaptive damping is deployed. Until then, the current adaptive damping measure is a practical and acceptable alternative for making sure that all individual non-linear solution processes, connected with all individual integration time steps in a GALILEO™ run, are driven to full and equal convergence accuracy.

XI. CONCLUSIONS

For large-scale routine applications of the continuous thermo-mechanical model in AREVA NP's fuel rod code GALILEO™, embedded systems of non-linear equations need to be solved many times over with high dependability and high computational efficiency. Within this context, a dedicated Broyden solver routine was developed for this particular purpose.

This paper presented a compact overview of the non-linear equations to be solved in GALILEO™'s continuous thermo-mechanical model, followed by a description of the final Broyden solver setup that proved adequate for optimizing robustness and computational efficiency. This final setup includes self-regulating mechanisms such as adaptive restarts and adaptive underrelaxation. These stability- and efficiency-enhancing mechanisms proved imperative for ensuring overall robustness and efficiency of the Broyden solver.

Verifications of the achieved solver properties, in terms of comparisons with simpler setups that lack adaptive measures, were presented in terms of associated observations on solver behavior.

REFERENCES

1. C. Garnier, H. Landskron, "GALILEO^{TM} Fuel Rod Performance Code - Theory Manual", internal report FS1-0004682, version 2.0 (2015).
2. N. Vioujard & al., "GALILEO^{TM} AREVA's Advanced Fuel Rod Performance Code and Associated Realistic Methodology", TOPFUEL Reactor Fuel Performance, Manchester, U.K. (2012).
3. A. Moal, V. Georgenthum, O. Marchand, "SCANAIR: A transient fuel performance code Part One: General modelling description", Nuclear Engineering and Design 280 150-171 (2014).
4. C. Petry, T. Helfer, "Advanced Mechanical Resolution in Cyrano3 Fuel Performance Code Using MFront Generation Tool ", TOPFUEL Reactor Fuel Performance, Zurich, Switzerland (2015).
5. Ph. Garcia & al. "Mono-Dimensional Mechanical Modelling of Fuel Rods Under Normal and Off-Normal Operating Conditions", Nuclear Engineering and Design 216 183-201 (2002).
6. K.-J. Bathe, "Finite Element Procedures", Prentice-Hall (1996).
7. J. Besson & al. "Non-Linear Mechanics of Materials", Springer (2010).
8. M. Abbas, "Quasi-static nonlinear algorithm (STAT_NON_LINE)", Code_Aster documentation, R5.03.01, 2013, http://www.code-aster.org/doc/v11/en/man_r/r5/r5.03.01.pdf.
9. T. Helfer & al.,"Introducing the Open-Source Mfront Code Generator: Application to Mechanical behaviors and Material Knowledge Management Within the PLEIADES Fuel Element Modelling Platform.", Computers & Mathematics with Applications, 70(5), 994-1023 (2015).
10. T. Helfer, "Etude de l'impact de la fissuration des combustibles nucléaires oxydes sur le comportement normal et incidentel des crayons combustibles", Ph.D. Ecole Centrale de Lyon, n^{o} EC2006-10 (2006).
11. C.G. Broyden, "A Class of Methods for Solving Nonlinear Simultaneous Equations", Mathematics of Computation 19 (92), pp. 577 - 593 (1965).
12. R. Burden, J. Faires, "Numerical Analysis (9^{th} edition)", Brooks/Cole CENGAGE Learning, International Edition (2011).
13. H. Fang, Y. Saad, "Two Classes of Multisecant Methods for Nonlinear Acceleration", *web article*, University of Minnesota (2007).
14. L. Knight, "Using Broyden Updates to Approximate the Jacobian in a Semi-Implicit Backward Differentiation Code" (1989).
15. B.W. Bader, R.P. Pawlowski, T.G. Kolda, "Robust Large-scale Parallel Nonlinear Solvers for Simulations", report SAND2005-6864, Sandia National Laboratory (2005).
16. H. Klie, M. Wheeler, "Nonlinear Krylov-Secant Solvers", *web article*, University of Texas (2011).