$$\underline{\underline{\bar{\mathbf{L}}}}^{it} = \mathcal{F}(\underline{\underline{\bar{\mathbf{L}}}}^{0}, \Delta\underline{\mathbf{u}}^{1}, \Delta\underline{\mathbf{f}}^{1}, \Delta\underline{\mathbf{u}}^{2}, \Delta\underline{\mathbf{f}}^{2}, ..., \Delta\underline{\mathbf{u}}^{it-1}, \Delta\underline{\mathbf{f}}^{it-1}), \tag{2.136}$$

where $\mathcal{F}(\bullet)$ denotes some function and the superscripts denote the iteration numbers.

In contrast to the BFGS method, the L-BFGS method does not use the iterative changes of all previous iterations, but only those of the n_{li} most recent iterations. Thus, in the general case that $it > n_{\mathrm{li}}$, the L-BFGS method writes the inverted Hessian approximation as follows (cf. Eq. (2.136)):

$$\underline{\underline{\bar{\mathbf{L}}}}^{it} = \mathcal{F}(\underline{\underline{\bar{\mathbf{L}}}}^{0}, \Delta\underline{\mathbf{u}}^{it-n_{\mathrm{li}}}, \Delta\underline{\mathbf{f}}^{it-n_{\mathrm{li}}}, \Delta\underline{\mathbf{u}}^{it-n_{\mathrm{li}}+1}, \Delta\underline{\mathbf{f}}^{it-n_{\mathrm{li}}+1}, ..., \Delta\underline{\mathbf{u}}^{it-1}, \Delta\underline{\mathbf{f}}^{it-1}). \tag{2.137}$$

The crucial difference with the BFGS approach is that the L-BFGS method does not explicitly compute, nor store the inverted Hessian approximation. Instead, it directly computes search direction $\underline{\mathbf{h}}$ according to the matrix-column product of Eq. (2.121) in terms of the intially assumed inverted Hessian approximation and the iterative changes of the estimated minimizer and of the associated gradient of the last n_{li} iterations. This can be performed relatively fast because the results of not all previous iterations are considered, but only those of the last n_{li} iterations. Consequently, only $2n_{\mathrm{it}}n_{\mathrm{var}}$ scalars need to be stored for the L-BFGS method, whereas n_{var}^2 must be stored for the BFGS method (where n_{var} denotes the number of unknowns in $\underline{\mathbf{u}}$).

Computing search direction $\underline{\mathbf{h}}$ according to Eq. (2.121) directly in terms of the initially assumed inverted Hessian approximation and the iterative changes of the minimizer and of the associated gradient of the last n_{li} iterations is achieved here by means of Alg. 2 [95]. The implementation of Alg. 2 is fairly simple to verify, because the predictions of the BFGS method and the L-BFGS method should be the same for the first n_{li} iterations.

2.3.3 Trust region method

The second true minimization algorithm investigated in this chapter is the trust region (TR) method [42, 43, 44, 39]. Whereas minimization methods such as the non-linear conjugate gradient method [96, 97] and quasi-Newton methods first compute a search direction and then calculate the stepsize, TR methods work the other way around. Hence, TR methods first need a (maximal) stepsize, and only then will they predict the correction to the current estimate of the minimizer.

The correction to the current estimate is computed by minimizing a quadratic approximation of the objective function, whilst safeguarding that the norm of the correction does not exceed the maximal stepsize. The maximal stepsize quantifies the size of the region in which the quadratic approximation can be trusted to be a suitable description of the exact objective function.

Depending on the obtained reduction of the objective function at the newly proposed estimate, the maximal stepsize for the next iteration is updated. The pink dash-dotted circles in Fig. 2.5 illustrate how the size of the trust region changes from one iteration to the next. The reduction obtained also determines whether the correction should or should not be used to update the current estimate of the minimizer.

To better explain TR methods, the computation of the correction to the current estimate is first considered. This computation is called the TR subproblem. Subsequently, the updating of the maximal stepsize for the next iteration is considered as well as whether or not the correction will be used to update the current estimate.

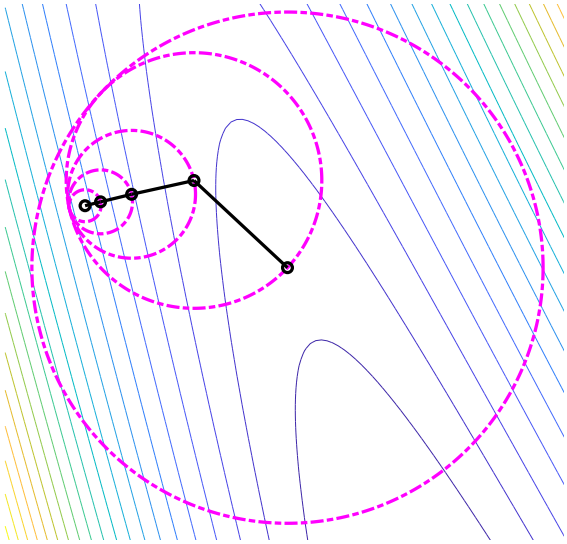

Figure 2.5: TR methods consider a trust region that is updated from one iteration to the next. The trust regions of different iterations are indicated by the pink dash-dotted circles. The black dots represent the estimate of the minimizer for different iterations. The colored lines depict isolines of an arbitrary objective function.

Trust region subproblem

The computation of correction $\underline{\mathbf{h}}^*$ to the current estimate of the minimizer, $\underline{\mathbf{u}}^{it}$, is based on the second-order Taylor expansion of the objective function, just like in Newton's method and quasi-Newton methods (cf. Eqs. (2.114) and (2.120)). Hence, TR methods write the computation of correction $\underline{\mathbf{h}}^*$ for given maximal stepsize R_{TR} as the following optimization problem:

$$\underline{\mathbf{h}}^* = \underset{\underline{\mathbf{h}}}{\operatorname{argmin}} \quad \Pi^{it} + \underline{\mathbf{h}}^T \cdot \underline{\mathbf{f}}^{it} + \frac{1}{2}\underline{\mathbf{h}}^T \cdot \underline{\underline{\mathbf{K}}}^{it} \cdot \underline{\mathbf{h}}, \tag{2.138}$$

$$\text{such that} \qquad \|\underline{\mathbf{h}}\| \leq R_{\mathrm{TR}}, \tag{2.139}$$

where

$$\|\underline{\mathbf{h}}\| = \sqrt{\underline{\mathbf{h}}^T \cdot \underline{\mathbf{h}}}\,. \tag{2.140}$$

It may be clear from Eq. (2.138) that the exact Hessian is used in this thesis. However, Hessian approximations that are updated during the iteration process as in quasi-Newton methods may also be used [39].

Although the optimization problem of Eq. (2.138) may seem similar to that of Newton's method of Eq. (2.114), it is rather different and not just because of the constraint in

Eq. (2.139). The reason is that Newton's method assumes that the minimizer is characterized by a zero gradient (Eq. (2.118)). However, this is true for any stationary point. Furthermore, Newton's method fails if a singular Hessian is encountered.

Often, the conjugate gradient variant of Steihaug and Toint [45, 46, 39] is used to solve the TR subproblem of Eqs. (2.138) and (2.139). The Steihaug-Toint conjugate gradient method is rather similar to the standard linear conjugate gradient method (to solve systems of linear equations as in Eq. (2.63)) [98]. The reason is that it first calculates a descent direction of the TR subproblem in a conjugate fashion (lines 1 and 18 to 25 in Alg. 3) and then calculates the stepsize in the search direction. Calculating the stepsize can most of the time be performed in a closed-form manner (line 10 in Alg. 3), similarly as in the standard linear conjugate gradient method to solve systems of linear equations [98], because the TR subproblem's objective function behaves quadratically in the search direction.

Nevertheless, the Steihaug-Toint conjugate gradient method varies from the standard linear method in three ways. First, the Steihaug-Toint variant requires to start at initial guess $\underline{\mathbf{h}} = \underline{\mathbf{0}}$ (line 1 of Alg. 3). This guarantees that $\|\underline{\mathbf{h}}\|$ increases for each iteration (see Fig. 2.6) and hence, it is easy to monitor if $\|\underline{\mathbf{h}}\|$ exceeds TR radius R_{TR}.

Second, if the minimizer of the TR subproblem's objective function (Eq. (2.138)) lies outside the trust region (case B in Fig. 2.6), the positive root of 1D quadratic function must be calculated to determine for which $\underline{\mathbf{h}}^*$ the current step crosses the TR boundary (lines 11 until 17 of Alg. 3). This location is then assumed to be the minimizer of the TR subproblem - according to Steihaug [46] and implemented here, but not according to Toint [45].

Third, the TR subproblem's objective function in the search direction may not have a minimum but a maximum. In this case, the stepsize in the current search direction must be taken as large as possible, until it crosses the TR boundary (case C in Fig. 2.6). Hence, the positive root of the same 1D quadratic function must be calculated to determine the associated stepsize (lines 3 to 9 of Alg. 3).

As may be noted in Alg. 3, the preconditioned variant of the Steihaug-Toint conjugate gradient method is also used here (Alg. 3 yields the unpreconditioned algorithm if $\underline{\underline{\mathbf{M}}}^{it} = \underline{\underline{\mathbf{I}}}$). Preconditioning reduces the number of conjugate gradient iterations. However, it often also affects the number of global TR iterations (denoted by the superscript it in the mathematical notation of this subsection). The reason for this is that the conjugate gradient method takes other directions if it is preconditioned. Consequently, if the TR boundary is crossed or a direction of negative curvature is encountered, the assumed minimizer (green stars in Fig. 2.6) will be located elsewhere compared to the unpreconditioned conjugate gradient method.

Preconditioners of the following format are often considered:

$$\underline{\underline{\mathbf{M}}}^{it} = \underline{\underline{\mathbf{\Lambda}}} \cdot \underline{\underline{\mathbf{\Lambda}}}^{cT} \approx \underline{\underline{\mathbf{K}}}^{it}. \tag{2.141}$$

where $\underline{\underline{\mathbf{\Lambda}}}$ denotes a lower triangular matrix of tensors and its transposed $\underline{\underline{\mathbf{\Lambda}}}^{cT}$ its upper triangular counterpart. Consequently, the original TR subproblem of Eqs. (2.138) and (2.139) is then rewritten with the help of:

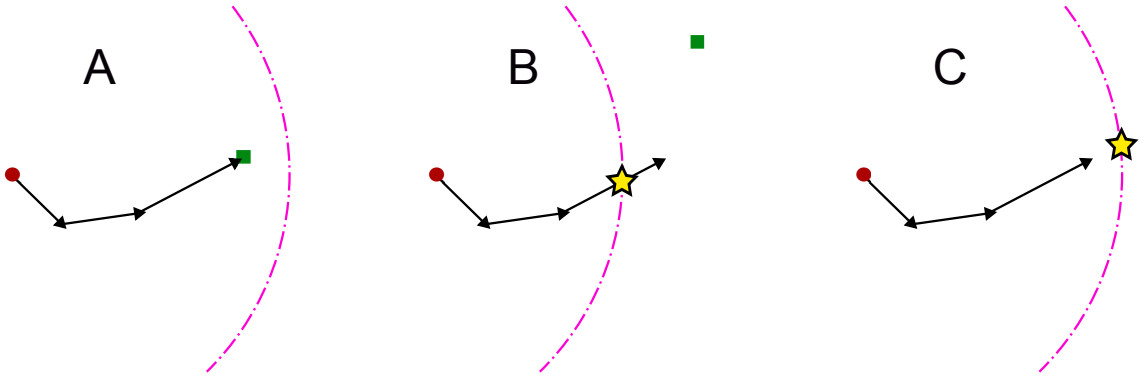

Figure 2.6: Artistic impression of the behaviour of the Steihaug-Toint conjugate gradient method for the three cases that can be encountered. Case A: the TR subproblem's minimizer is located within the TR boundary, case B: the TR subproblem's minimizer is located outside the TR boundary, case C: a direction with negative curvature is encountered during the conjugate gradient process. Red dots: starting points ($\underline{\mathbf{h}} = \underline{\mathbf{0}}$), pink dashed-dotted circles: TR boundaries, black arrows: updates $\underline{\mathbf{p}}$ for $\underline{\mathbf{h}}$, green squares: minimizers of Eq. (2.138), yellow stars: assumed minimizer of the TR subproblem if $\underline{\mathbf{p}}$ crosses the TR boundary (case B), or if $\underline{\mathbf{p}}$ is a direction of negative curvature (case C). This is only an artistic impression because, for a 2D problem as sketched here, the conjugate gradient method would converge in a maximum of two iterations.

$$\underline{\mathbf{h}} = \underline{\underline{\mathbf{\Lambda}}}^{-cT} \cdot \bar{\underline{\mathbf{h}}}, \tag{2.142}$$

as:

$$\bar{\underline{\mathbf{h}}}^* = \underset{\bar{\underline{\mathbf{h}}}}{\operatorname{argmin}} \quad \Pi^{it} + \bar{\underline{\mathbf{h}}}^T \cdot \bar{\underline{\mathbf{f}}}^{it} + \frac{1}{2} \bar{\underline{\mathbf{h}}}^T \cdot \bar{\underline{\underline{\mathbf{K}}}}^{it} \cdot \bar{\underline{\mathbf{h}}}, \tag{2.143}$$

$$\text{such that} \qquad \|\bar{\underline{\mathbf{h}}}\| \leq R_{\text{TR}}, \tag{2.144}$$

where $\bar{\underline{\mathbf{h}}}$ denotes the preconditioned correction and

$$\bar{\underline{\mathbf{f}}}^{it} = \underline{\underline{\mathbf{\Lambda}}}^{-1} \cdot \underline{\mathbf{f}}^{it}, \tag{2.145}$$

$$\bar{\underline{\underline{\mathbf{K}}}}^{it} = \underline{\underline{\mathbf{\Lambda}}}^{-1} \cdot \underline{\underline{\mathbf{K}}}^{it} \cdot \underline{\underline{\mathbf{\Lambda}}}^{-cT}. \tag{2.146}$$

It may be observed in Alg. 3 that only preconditioner $\underline{\underline{\mathbf{M}}}^{it}$ is required and not necessarily the factorization of Eq. (2.141). Nevertheless, this thesis considers the incomplete Cholesky factorization of the Hessian, whilst [31] considers the full Cholesky factorization. No fill-in is used nor a drop tolerance. Thus, $\underline{\underline{\mathbf{\Lambda}}}$ and $\underline{\underline{\mathbf{\Lambda}}}^T$ in Eq. (2.141) denote the lower and upper triangular forms of the Hessian, respectively.

In case of incomplete Cholesky preconditioning, the preconditioner $\underline{\underline{\mathbf{M}}}^{it}$ is not guaranteed to be invertible. Therefore, the implementation of this thesis monitors if the preconditioner

is singular, in which case it alters the Hessian's diagonal and recomputes the incomplete Cholesky factorization. This procedure is repeated as often as necessary.

Similar to [31] (and for instance standard available in MATLAB), if the preconditioner is singular, this thesis alters the diagonal of the Hessian by inflating its diagonal components according to:

$$\operatorname{diag}\left(\underline{\underline{\mathbf{K}}}^{it}\right) = \left(1 + 10^{i-8}\right) \operatorname{diag}\left(\underline{\underline{\mathbf{K}}}^{it}\right), \tag{2.147}$$

where i refers to the i^{th} time that the Hessian's diagonal of the current iteration is inflated and diag($\bullet$) refers to the diagonal of the second-order tensors on the diagonal of the matrix. As [31] mentions, this procedure will eventually provide an invertible triangular form if the diagonal components of the Hessian are positive.

Where this thesis differs from [31], however, is that this thesis' examples push the solver to such limits that negative diagonal components are occasionally encountered. In such scenarios, the negative diagonal components are simply made positive before they are potentially inflated. Also different from [31] is that this thesis recomputes the preconditioner for each iteration, whereas [31] only does so depending on the preconditioner's performance for the previous TR iteration (quantified by the number of conjugate gradient iterations of the previous global TR iteration).

Updating in trust region methods

Once the TR subproblem of Eqs. (2.143) and (2.144) is solved, $\underline{\mathbf{h}}^*$ is known. The next questions the TR algorithm must answer is whether correction $\underline{\mathbf{h}}^*$ is good enough to be used as an update of the current estimate $\underline{\mathbf{u}}^{it}$, and whether or not the second-order Taylor approximation of Eq. (2.138) has reflected the exact objective function sufficiently accurately in the TR (characterized by the radius of TR R_{TR}).

For both decisions, TR methods consider the ratio between the reduction of the exact objective function generated by the correction $\underline{\mathbf{h}}^*$ and the reduction that is expected based on the approximation of Eq. (2.138). The ratio between the two, ρ, can thus be expressed as follows:

$$\rho = \frac{\Pi^{it} - \Pi^{it+h^*}}{\bar{\Pi}^{it}(\underline{\mathbf{0}}) - \bar{\Pi}^{it}(\underline{\mathbf{h}}^*)}, \tag{2.148}$$

where Π^{it} denotes the exact objective function at $\underline{\mathbf{u}}^{it}$ as provided in Eq. (2.115) and Π^{it+h^*} denotes the exact objective function at $\underline{\mathbf{u}}^{it} + \underline{\mathbf{h}}^*$:

$$\begin{aligned}\Pi^{it+h^*} = \Pi_{\text{mat/u}}\left(\underline{\mathbf{u}}^{it} + \underline{\mathbf{h}}^*, \underline{\underline{\zeta}}_{\text{prev}}, \underline{\underline{\zeta}}^{*it+h^*}\right) \\ - \Pi_{\text{con}}\left(\underline{\mathbf{u}}^{it} + \underline{\mathbf{h}}^*, \underline{\underline{\zeta}}^{*it+h^*}(\underline{\mathbf{u}}^{it} + \underline{\mathbf{h}}^*)\right) - \underline{\mathbf{f}}_{\text{ext}}^T \cdot \left(\underline{\mathbf{u}}^{it} + \underline{\mathbf{h}}^*\right).\end{aligned} \tag{2.149}$$

Furthermore, $\bar{\Pi}^{it}$ denotes the objective function approximation used in the TR subproblem of Eq. (2.138).

Algorithm 3 Preconditioned Steihaug-Toint conjugate gradient method.

INPUT: R_{TR}, $\underline{\mathbf{f}}^{it}$, $\underline{\underline{\mathbf{K}}}^{it}$, $\underline{\underline{\mathbf{M}}}^{it}$

$\underline{\mathbf{r}} = -\underline{\mathbf{f}}^{it}$, Solve $\underline{\underline{\mathbf{M}}}^{it} \cdot \underline{\mathbf{p}} = \underline{\mathbf{r}}$ for $\underline{\mathbf{p}}$, $\underline{\mathbf{q}} = \underline{\mathbf{p}}$, $\underline{\mathbf{h}}^* = \underline{\mathbf{0}}$
while do
  if $\underline{\mathbf{p}}^T \cdot \underline{\underline{\mathbf{K}}}^{it} \cdot \underline{\mathbf{p}} \leq 0$ **then**
    Raise *flag* that TR boundary is reached
    $a = \underline{\mathbf{p}}^T \cdot \underline{\underline{\mathbf{M}}}^{it} \cdot \underline{\mathbf{p}}$, $b = 2\underline{\mathbf{p}}^T \cdot \underline{\underline{\mathbf{M}}}^{it} \cdot \underline{\mathbf{h}}^*$, $c = \underline{\mathbf{h}}^{*T} \cdot \underline{\underline{\mathbf{M}}}^{it} \cdot \underline{\mathbf{h}}^* - R_{\mathrm{TR}}^2$
    $\alpha = \frac{-b+\sqrt{b^2-4ac}}{2a}$
    $\underline{\mathbf{h}}^* = \underline{\mathbf{h}}^* + \alpha\,\underline{\mathbf{p}}$
    BREAK
  end if
  $\alpha = \frac{\underline{\mathbf{r}}^T \cdot \underline{\mathbf{q}}}{\underline{\mathbf{p}}^T \cdot \underline{\underline{\mathbf{K}}}^{it} \cdot \underline{\mathbf{p}}}$
  if $(\underline{\mathbf{h}}^* + \alpha\underline{\mathbf{p}})^T \cdot \underline{\underline{\mathbf{M}}}^{it} \cdot (\underline{\mathbf{h}}^* + \alpha\underline{\mathbf{p}}) > R_{\mathrm{RT}}^2$ **then**
    Raise *flag* that TR boundary is reached
    $a = \underline{\mathbf{p}}^T \cdot \underline{\underline{\mathbf{M}}}^{it} \cdot \underline{\mathbf{p}}$, $b = 2\underline{\mathbf{p}}^T \cdot \underline{\underline{\mathbf{M}}}^{it} \cdot \underline{\mathbf{h}}^*$, $c = \underline{\mathbf{h}}^{*T} \cdot \underline{\underline{\mathbf{M}}}^{it} \cdot \underline{\mathbf{h}}^* - R_{\mathrm{TR}}^2$
    $\alpha = \frac{-b+\sqrt{b^2-4ac}}{2a}$
    $\underline{\mathbf{h}}^* = \underline{\mathbf{h}}^* + \alpha\,\underline{\mathbf{p}}$
    BREAK
  end if
  $\underline{\mathbf{h}}^* = \underline{\mathbf{h}}^* + \alpha\,\underline{\mathbf{p}}$
  $\phi = \underline{\mathbf{r}}^T \cdot \underline{\mathbf{p}}$
  $\underline{\mathbf{r}} = \underline{\mathbf{r}} - \alpha\,\underline{\underline{\mathbf{K}}}^{it} \cdot \underline{\mathbf{p}}$
  if $\|\underline{\mathbf{r}}\| < \max\left(10^{-15}, 10^{-5}\|\underline{\mathbf{f}}^{it}\|\right)$ **then**
    BREAK
  end if
  Solve $\underline{\underline{\mathbf{M}}}^{it} \cdot \underline{\mathbf{q}} = \underline{\mathbf{r}}$ for $\underline{\mathbf{q}}$
  $\underline{\mathbf{p}} = \underline{\mathbf{q}} + \frac{\underline{\mathbf{r}}^T \cdot \underline{\mathbf{q}}}{\phi}\,\underline{\mathbf{p}}$
end while

OUTPUT: $\underline{\mathbf{h}}^*$, *flag* (indicating whether or not TR boundary is reached)

Algorithm 4 Updating in the TR method of this thesis.

INPUT: $\underline{\mathbf{h}}^*$, R_{TR}, $R_{\mathrm{TR/max}}$, Π^{it}, $\underline{\mathbf{f}}^{it}$, $\underline{\underline{\mathbf{K}}}^{it}$, *flag* (indicating whether or not TR boundary was reached)

Evaluate $\underline{\mathbf{f}}^{it+h^*}$ for $\underline{\mathbf{u}}^{it} + \underline{\mathbf{h}}^*$
if $\|\underline{\mathbf{f}}^{it+h^*}\| = \mathrm{NaN}$ **then**
    $\rho = 0$
else
    $\rho = \frac{\underline{\mathbf{h}}^{*T} \cdot (\underline{\mathbf{f}}^{it} + \underline{\mathbf{f}}^{it+h^*})}{2\underline{\mathbf{h}}^{*T} \cdot \underline{\mathbf{f}}^{it} + \underline{\mathbf{h}}^{*T} \cdot \underline{\underline{\mathbf{K}}}^{it} \cdot \underline{\mathbf{h}}^*}$
end if
if $\rho < 0.25$ **then**
    $\underline{\mathbf{u}}^{it+1} = \underline{\mathbf{u}}^{it}$
    $R_{\mathrm{TR}} = 0.25 R_{\mathrm{TR}}$
else
    $\underline{\mathbf{u}}^{it+1} = \underline{\mathbf{u}}^{it} + \underline{\mathbf{h}}^*$
    if $\rho > 0.75$ **and** 'TR boundary was reached' **then**
        $R_{\mathrm{TR}} = \min(2R_{\mathrm{TR}}, R_{\mathrm{TR/max}})$
    end if
end if

OUTPUT: $\underline{\mathbf{u}}^{it+1}$, R_{TR}

As the objective function is substantially more easily polluted by noise due to machine precision than the gradient, the expression of Eq. (2.148) cannot be used. For this purpose, this thesis borrows the formulation for ρ of [31], which assumes that the gradient in direction $\underline{\mathbf{h}}^*$ scales similarly to the objective function in direction $\underline{\mathbf{h}}^*$. Thus, the following expression is used here:

$$\rho = \frac{\underline{\mathbf{h}}^{*T} \cdot (\underline{\mathbf{f}}^{it} + \underline{\mathbf{f}}^{it+h^*})}{2\underline{\mathbf{h}}^{*T} \cdot \underline{\mathbf{f}}^{it} + \underline{\mathbf{h}}^{*T} \cdot \underline{\underline{\mathbf{K}}}^{it} \cdot \underline{\mathbf{h}}^*}, \tag{2.150}$$

where the denominator is replaced with the help of Eq. (2.138).

Alg. 4 clearly demonstrates that the ratio ρ is used to both decide whether or not estimate $\underline{\mathbf{u}}^{it}$ must be updated with correction $\underline{\mathbf{h}}^*$, and if the TR radius R_{TR} must be decreased for the next iteration, kept the same or increased. $R_{\mathrm{TR/max}}$ in Alg. 4 denotes the maximal TR radius as defined by the user. The full TR algorithm (not shown) also requires the user to define an initial value for TR radius R_{TR}.

2.4 Gregory patches

One point that has not yet been discussed is the leader-surface description. The current section sheds details on this. In other words, the current section discusses the formulation of $\mathbf{X}_{\mathrm{lead}}$ and $\mathbf{n}_{\mathrm{lead}}$ in Eqs. (2.11) and (2.12), as a function of the two parametric surface coordinates in $\underline{\xi}$.

This thesis considers FE vertices on the deformable body's surface as the follower-vertices