3.4 Iterative Methods for Nonlinear Systems

Our original problem (1.1) is as mentioned before potentially nonlinear, but so far we have only been discussing ways of how to discretise linear problems. Let us therefore now undertake a short excursion of how to solve non-linear problems, which in each iteration step will include solving linearisations that are of the beforementioned type. We assume that we can write the nonlinear *problem* in the subsequent form and are therefore interested in solving *problems* of the following type

$$\text{Find } s \in \Omega \subset \mathbb{R}^m \text{ such that } J(s) = 0. \tag{3.27}$$

for some $J : \Omega \subset \mathbb{R}^m \to \mathbb{R}_{\geq 0}$. Since this describes a very broad class of problems there are of course many different approaches of how to tackle this question. Here we will introduce two well-known possibilities,gradient descent methods and Newton's method, as well as a combination of the two in the form a trust region method.

3.4.1 Gradient Descent Methods

The method of gradient descent is a relatively simple and inexpensive iterative optimisation algorithm used to find a local minimum of a function J. We only have to require that the function J is differentiable in a neighbourhood of each current iterate s_k. After an initial guess s_0 is chosen, one takes a step in the direction of the negative gradient of J at s_0, that is the direction of steepest descent. The iteration then looks as follows

$$s_{k+1} = s_k + \alpha_k(-\nabla J(s_k)) \tag{3.28}$$

If the scaling parameter $\alpha_k > 0$ is chosen sufficiently small, we know that $J(s_{k-1}) \geq J(s_k) \geq J(s_{k+1})$. If $||\nabla J(s_k)|| = 0$ we have found a local minimum and hence $s_{k+1} = s_k$. There are a number of strategies that try to select a suitable value for α_k, one of them is for example a line search algorithm using Wolfe's condition [source].

Under the assumption that $J \in C^1(\Omega)$, bounded and convex and particular choices for the α_k, e.g. using an above mentioned line search, the method is guaranteed to converge to a local minimum which is due to the convexity of J the unique global minimiser. However the speed of convergence can be extremely slow for *ill-conditioned* problems [source]. Gradient Descent represents a first-order Taylor approximation of J in s_k and gives an updated solution based on this local linear model. A more sophisticated approach than this is for example Newton's method which uses a second order Taylor approximation.

3.4.2 Newton's Method

[source Briggs] It is one of the most well-known and most commonly used methods to solve non-linear problems of the above type. Let $J \in C^2(\Omega)$, and hence if differentiate both sides of the equation $J(s) = 0$, we obtain $\nabla J(s) = [0, 0, ..., 0]^T$, for which we would like to determine the unknown root s. We consider a Taylor series expansion for an initial guess s_0:

$$\nabla J(s_0 + h) = \nabla J(s_0) + \nabla^2 J(s_0) \cdot h + o(||h||^2), \quad s_0 \in \Omega. \tag{3.29}$$

If we now neglect the higher order terms, setting $\nabla J(s_0 + h) = 0$ and replacing it by its first order Taylor approximation $\nabla J(s_0) + \nabla^2 J(s_0) \cdot h$, which we can then solve for h, under the assumption that $\nabla^2 J(s_0)$ is non-singular and use the result to update our initial guess s_0. One ends up the with iteration

$$s_{k+1} = s_k - [\nabla^2 J(s_k)]^{-1} \nabla J(s_k). \tag{3.30}$$

In the case of J being convex and a few additional conditions one can achieve a quadratic rate of convergence for Newton's method compared to a linear one for the method of gradient descent. However except for the one-dimensional case, it is usually very hard or impossible to know if these conditions are actually fulfilled [source]. And in addition to the gradient of J, one also has to compute the inverse of the Hessian of J in each iteration. For larger systems this is in most cases a rather difficult and computationally expensive problem, which in our case we will try to tackle using a multigrid method [see chapter 5].

Hence, a quadratic approximation to find a minimiser only makes sense for a locally convex neighbourhood, otherwise the Newton iteration might not lead to a decrease but instead an increase in energy as it might take an iteration step towards a local maximum. Therefore in order to make use of the faster rate of convergence in convex neighbourhoods of J it makes sense to use a Newton iteration, while in non-convex neighbourhoods it can be preferrable to use a gradient descent step as it is guaranteed to not increase the value of the functional. One option to combine these two is by using a trust region algorithm which has an additional important parameter, the so-called trust region radius and will be introduced in the following section.

3.4.3 Trust Region Methods

Trust region methods is a generic term that comprises globalisation strategies to approximate minima of potentially non-convex optimisation problems. The objective function J is approximated using a model function in a subset of the domain, the so-called trust region. If the model function seems to be a good local fit to the function, the size of the region is expanded, if it is not, the size of the region is reduced. The fit of the model is assessed by comparing the ratio ρ_k of the expected reduction of the objective function by the model and the actual reduction of the function.

A typical iteration step k can look as follows. We have a current trust region that is usually defined as a spherical area of radius Δ_k and a model function $m_k(p)$ that is supposed to locally approximate J and where p describes the update to the current solution, that is the new step to be taken. We therefore want to solve for p, hence we minimise over p within the trust region radius to obtain a solution p_k and can thus determine ρ_k, the ratio of the expected compared to the actual reduction. Depending on its value and the parameter thresholding we either reduce Δ_k if the approximation does not seem like a good fit, and then solve m_k again for p with a smaller Δ_{k+1}. Otherwise we either enlarge it, if $||p_k|| = \Delta_k$, i.e. it is maximal or else leave it the same. If ρ_k is not too small we compute the new solution $s_{k+1} = s_k + p_k$.

There are many options what to choose as a model function. Among the simplest approaches is the so-called Cauchy point calculation that stems from gradient descent, other popular ones include steihaug's method or the so-called dogleg method [source]. The former uses a conjugate gradient method for a quadratic model function and the later includes a mixture of using information of the first and second derivative of J, which is what we used and which will be explained in more detail in the chapter on the implementation [see 6.6.3].