Technische Universität Braunschweig

Partitioned Methods for Multifield Problems

Rang, 17.5.2017

Technische Universität Braunschweig

Newton's method

Nonlinear problem:

$$\mathbf{f}(\mathbf{x}, \mathbf{y}) = 0,$$
$$\mathbf{g}(\mathbf{x}, \mathbf{y}) = 0$$

Newton's method:

$$\begin{pmatrix} \mathbf{x}^{(\nu+1)} \\ \mathbf{y}^{(\nu+1)} \end{pmatrix} =$$
$$\begin{pmatrix} \mathbf{x}^{(\nu)} \\ \mathbf{y}^{(\nu)} \end{pmatrix} - \begin{pmatrix} \mathbf{f}_\mathbf{x}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) & \mathbf{f}_\mathbf{y}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) \\ \mathbf{g}_\mathbf{x}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) & \mathbf{g}_\mathbf{y}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{f}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) \\ \mathbf{g}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) \end{pmatrix}$$

Newton's method

- Let $\Delta \mathbf{x}^{(\nu+1)} := \mathbf{x}^{(\nu+1)} - \mathbf{x}^{(\nu)}$ and $\Delta \mathbf{y}^{(\nu+1)} := \mathbf{y}^{(\nu+1)} - \mathbf{y}^{(\nu)}$.
- A simple reformulation leads to the linear system

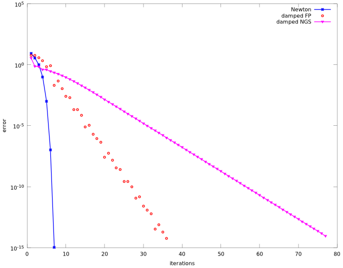
$$\begin{pmatrix} \mathbf{f}_{\mathbf{x}}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) & \mathbf{f}_{\mathbf{y}}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) \\ \mathbf{g}_{\mathbf{x}}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) & \mathbf{g}_{\mathbf{y}}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) \end{pmatrix} \begin{pmatrix} \Delta \mathbf{x}^{(\nu+1)} \\ \Delta \mathbf{y}^{(\nu+1)} \end{pmatrix} = - \begin{pmatrix} \mathbf{f}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) \\ \mathbf{g}(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}) \end{pmatrix}.$$

Example

- Consider the problem

$$0 = x^2 + y^2 - 25$$
$$0 = y - x - 1$$

- starting values $x_0 = y_0 = 1$



Partititioned methods

Nonlinear problem:

$$\mathbf{f}(\mathbf{x}, \mathbf{y}) = 0,$$
$$\mathbf{g}(\mathbf{x}, \mathbf{y}) = 0$$

- Block–Jacobi method
- Block–Gauß–Seidel method
- Block–SOR method
- Block–Newton method
- Block Approximated Newton method (BAN method)

The Block–Jacobi method

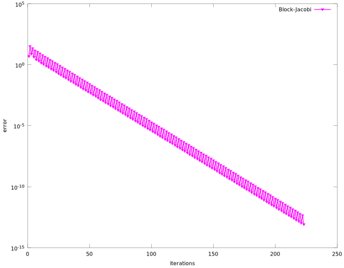
1. Set $\nu := 0$ and choose an initial guess for $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$.
2. Compute $\mathbf{x}^{(\nu+1)}$ by solving
$$0 = \mathbf{f}\left(\mathbf{x}^{(\nu+1)}, \mathbf{y}^{(\nu)}\right).$$
3. Compute $\mathbf{y}^{(\nu+1)}$ by solving
$$0 = \mathbf{g}\left(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu+1)}\right).$$
4. Set $\nu := \nu + 1$.
5. Compute the residuum
$$r^{(\nu)} := \left\|\mathbf{f}\left(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}\right)\right\| + \left\|\mathbf{g}\left(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}\right)\right\|.$$
6. If $|r^{(\nu)}|$ is sufficiently small, then stop. Otherwise compute $\mathbf{x}^{(v+1)}$ and $\mathbf{y}^{(v+1)}$.

Example

- Consider the problem

$$0 = x^2 + y^2 - 25$$
$$0 = y - x - 1$$

- starting values $x_0 = y_0 = 1$



The Block–Gauß–Seidel method

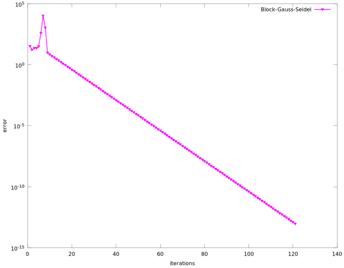
1. Set $\nu := 0$ and choose an initial guess for $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$.
2. Compute $\mathbf{x}^{(\nu+1)}$ by solving
$$0 = \mathbf{f}\left(\mathbf{x}^{(\nu+1)}, \mathbf{y}^{(\nu)}\right).$$
3. Compute $\mathbf{y}^{(\nu+1)}$ by solving
$$0 = \mathbf{g}\left(\mathbf{x}^{(\nu+1)}, \mathbf{y}^{(\nu+1)}\right).$$
4. Set $\nu := \nu + 1$.
5. Compute the residuum
$$r^{(\nu)} := \left\|\mathbf{f}\left(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}\right)\right\| + \left\|\mathbf{g}\left(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}\right)\right\|.$$
6. If $|r^{(\nu)}|$ is sufficiently small, then stop. Otherwise compute $\mathbf{x}^{(v+1)}$ and $\mathbf{y}^{(v+1)}$.

Example

- Consider the problem

$$0 = x^2 + y^2 - 25$$
$$0 = y - x - 1$$

- starting values $x_0 = y_0 = 1$



Block–SOR method

Extension of Block–Gauß–Seidel method:

1. Set $\nu := 0$ and choose an initial guess for $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$.
2. Compute $\mathbf{x}^{(\nu+1)}$ by solving
$$0 = \mathbf{f}\left(\mathbf{x}^{(\nu+1)}, \mathbf{y}^{(\nu)}\right).$$
3. Set $\tilde{\mathbf{x}}^{(\nu+1)} := \mathbf{x}^{(\nu)} + \omega(\mathbf{x}^{(\nu+1)} - \mathbf{x}^{(\nu)})$.
4. Compute $\mathbf{y}^{(\nu+1)}$ by solving
$$0 = \mathbf{g}\left(\tilde{\mathbf{x}}^{(\nu+1)}, \mathbf{y}^{(\nu+1)}\right).$$
5. Set $\nu := \nu + 1$.
6. Compute the residuum
$$r^{(\nu)} := \left\|\mathbf{f}\left(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}\right)\right\| + \left\|\mathbf{g}\left(\mathbf{x}^{(\nu)}, \mathbf{y}^{(\nu)}\right)\right\|.$$
7. If $|r^{(\nu)}|$ is sufficiently small, then stop. Otherwise compute $\mathbf{x}^{(v+1)}$
$+1)$

The Block-Newton method

First we apply Newton's method on our nonlinear system

$$0 = \mathbf{f}(\mathbf{x}, \mathbf{y}), \tag{1}$$

$$0 = \mathbf{g}(\mathbf{x}, \mathbf{y}). \tag{2}$$

We obtain the linear system

$$\begin{pmatrix} \mathbf{f}_x(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) & \mathbf{f}_y(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) \\ \mathbf{g}_x(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) & \mathbf{g}_y(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) \end{pmatrix} \begin{pmatrix} \Delta\mathbf{x}^{(k+1)} \\ \Delta\mathbf{y}^{(k+1)} \end{pmatrix} = - \begin{pmatrix} \mathbf{f}(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) \\ \mathbf{g}(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) \end{pmatrix}, \tag{3}$$

where $\mathbf{f}_x$ and so on are the Jacobi matrices and $\Delta\mathbf{x}^{(k+1)} := \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}$.

The Block-Newton method

Apply one Gauß-step, i.e. we resolve the first equation of (3) w.r.t. $\Delta\mathbf{x}^{(k+1)}$, i.e.

$$\Delta\mathbf{x}^{(k+1)} = -\mathbf{f}_x^{-1}(\mathbf{f}_y\Delta\mathbf{y}^{(k+1)} + \mathbf{f})$$

and insert this result into the second equation, i.e.

$$(\mathbf{g}_y - \mathbf{g}_x\mathbf{f}_x^{-1}\mathbf{f}_y)\Delta\mathbf{y}^{(k+1)} = \mathbf{g}_x\mathbf{f}_x^{-1}\mathbf{f} - \mathbf{g}.$$

For abbreviation we set

$$S := \mathbf{g}_y - \mathbf{g}_x\underbrace{\mathbf{f}_x^{-1}\mathbf{f}_y}_{=:C}, \quad \mathbf{p} := \mathbf{f}_x^{-1}\mathbf{f}.$$

The matrix S is often called Schur complement.

The Block-Newton method

1. Compute $\mathbf{p}$, i.e. solve $\mathbf{f}_x\mathbf{p} = \mathbf{f}$ for $\mathbf{p}$.
2. Compute $C = \mathbf{f}_x^{-1}\mathbf{f}_y$, i.e. solve the matrix equation $\mathbf{f}_x C = \mathbf{f}_y$ for C.
3. Compute the Schur complement $S = \mathbf{g}_y - \mathbf{g}_x C$.
4. Compute the modified right-hand side $\mathbf{g}_x\mathbf{p} - \mathbf{g} =: -\tilde{\mathbf{g}}$.
5. Solve $S\Delta\mathbf{y} = -\tilde{\mathbf{g}}$ for $\Delta\mathbf{y}$.
6. Compute $\Delta\mathbf{x} = -(\mathbf{p} + C\Delta\mathbf{y})$.

Remarks

- This algorithm has the disadvantage that the computation of C is rather expensive.
- Moreover a lot matrix computation have to be done and finally the cross derivatives are often not available.

The BAN method

- In the following we set $\mathbf{x}^c = \mathbf{x}^{(k)}$ and $\mathbf{y}^c = \mathbf{y}^{(k)}$ for simplification.
- Let us assume in the follwing that we have two codes. The first code solves (1) and knows only $\mathbf{x}$ whereas the second code solves (2) and knows only $\mathbf{y}$.
- It is of course clear that the solver for (1) does not know the derivative w.r.t. $\mathbf{y}$ since $\mathbf{y}$ is a constant variable. In the same way the solver for (2) does not know the derivative w.r.t. $\mathbf{x}$. Hence the derivatives $\mathbf{f}_y$ and $\mathbf{g}_x$ are in general unknwon and should be approximated.

The BAN method

In the first step the linear equations $\mathbf{f}_x(\mathbf{x}^c)\mathbf{p} = \mathbf{f}$ should be solved to get an approximation for $\mathbf{p}$. It follows

$$\mathbf{x}^c - \mathbf{p} = \mathbf{x}^c - \mathbf{f}_x(\mathbf{x}^c, \mathbf{y}^c)^{-1}\mathbf{f}(\mathbf{x}^c, \mathbf{y}^c). \tag{4}$$

The right-hand side of (4) is a Newton approximation in x-direction of $\mathbf{f}(\mathbf{x}, \mathbf{y}^c) = 0$ at $\mathbf{x}^c$ for fixed $\mathbf{y}^c$, i.e.

$$\mathbf{x} \approx \mathbf{x}^c - \mathbf{p}$$

is a second order approximation. Our value $\mathbf{p}$ can be approximated by

$$\mathbf{p} \approx \mathbf{x}^c - \mathbf{x},$$

but $\mathbf{x}$ is not available.

The BAN method

Therefore it can approximated by the given iteration

$$\mathbf{x}^{(j+1)} = \mathbf{F}(\mathbf{x}^{(j)}, \mathbf{y}^c), \quad j = 0, 1, \ldots, \quad \mathbf{x}^{(0)} = \mathbf{x}^c, \tag{5}$$

such that

$$\mathbf{p} \approx \mathbf{x}^c - \mathbf{x}^{(j)}, \quad j \geqslant 1. \tag{6}$$

The BAN method

The matrix $C = \mathbf{f}_x^{-1}\mathbf{f}_y$ is only needed in matrix-vector operations. The cross derivative $\mathbf{f}_y$ is approximated with finite differences and the derivative $\mathbf{f}_x$ is used in a Newton approximation. Let $\mathbf{w}$ be some vector. Then

$$C\mathbf{w} \approx \frac{1}{h}\mathbf{f}_x^{-1}(\mathbf{x}^c, \mathbf{y}^c)\underbrace{\left(\mathbf{f}(\mathbf{x}^c, \mathbf{y}^c + h\mathbf{w}) - \mathbf{f}(\mathbf{x}^c, \mathbf{y}^c)\right)}_{\approx h\mathbf{f}_y\mathbf{w}}. \tag{7}$$

The right-hand side of (7) is a Newton step to solve

$$0 = \mathbf{f}(\mathbf{x}^c, \mathbf{y}^c + h\mathbf{w}) - \mathbf{f}(\mathbf{x}^c, \mathbf{y}^c)$$

in $\mathbf{x}$ at $\mathbf{x}^{(k)}$ for fixed $\mathbf{y}^{(k)}$.

The BAN method

As in the last step this solution can be found by the iteration

$$\mathbf{x}^{(j+1)} = \mathbf{F}(\mathbf{x}^{(j)}, \mathbf{y}^{(k)}) + \mathbf{f}(\mathbf{x}^c, \mathbf{y}^c + h\mathbf{w}), \quad j = 0, 1, \ldots, \quad \mathbf{x}^{(0)} = \mathbf{x}^c, \tag{8}$$

such that

$$C\mathbf{w} \approx \frac{\mathbf{x}^{(j)} - \mathbf{x}^c}{h}, \quad j \geqslant 1.$$

Approximate $\tilde{\mathbf{g}} := \mathbf{g} - \mathbf{g}_x\mathbf{p}$ as follows. A taylor expansion of $\mathbf{g}(\mathbf{x}^c - \mathbf{p}, \mathbf{y}^c)$ leads to

$$\mathbf{g}(\mathbf{x}^c - \mathbf{p}, \mathbf{y}^c) = \mathbf{g}(\mathbf{x}^c, \mathbf{y}^c) - \mathbf{g}_x(\mathbf{x}^c, \mathbf{y}^c)\mathbf{p} = \tilde{\mathbf{g}}(\mathbf{x}^c, \mathbf{y}^c).$$

The BAN method

As the matrix C the Schur complement S is only needed in matrix-vector operations. We have

$$\begin{aligned} hS\mathbf{w} &= h\left(\mathbf{g}_y(\mathbf{x}^c, \mathbf{y}^c) - \mathbf{g}_x(\mathbf{x}^c, \mathbf{y}^c)C\right)\mathbf{w} \\ &\approx \mathbf{g}(\mathbf{x}^c - hC\mathbf{w}, \mathbf{y}^c + h\mathbf{w}) - \mathbf{g}(\mathbf{x}^c, \mathbf{y}^c), \end{aligned}$$

which is the taylor expansion of the first equality.

The BAN method

And finally we should think about the approximation of $\Delta\mathbf{x}$. Let us assume that we know $\Delta\mathbf{y}$. Then we can compute $\Delta\mathbf{x}$ from the equation

$$f(\mathbf{x}^c + \Delta\mathbf{x}, \mathbf{y}^c + \Delta\mathbf{y}) = 0.$$

As in the step for the computation of **p** we can use the iteration

$$\mathbf{x}^{(j+1)} = \mathbf{F}(\mathbf{x}^{(j)}, \mathbf{y}^c + \Delta\mathbf{y}), \quad j = 0, 1, \ldots, \quad \mathbf{x}^{(0)} = \mathbf{x}^c. \tag{9}$$

Finally we have the approximation

$$\Delta\mathbf{x} \approx \mathbf{x}^c - \mathbf{x}^{(j)}, \quad j \geqslant 1. \tag{10}$$

The BAN method

1. Compute $\mathbf{p} \approx \mathbf{x}^c - \mathbf{x}^{(j)}, \quad j \geqslant 1$, where $\mathbf{x}^{(j)}$ is given by (3), i.e.
$$\mathbf{x}^{(j+1)} = \mathbf{F}(\mathbf{x}^{(j)}, \mathbf{y}^c), \quad j = 0, 1, \dots, \quad \mathbf{x}^{(0)} = \mathbf{x}^c.$$
2. Compute the modified right-hand side $\tilde{\mathbf{g}}$ by
$$\tilde{\mathbf{g}} = \mathbf{g}(\mathbf{x}^c - \mathbf{p}, \mathbf{y}^c).$$
3. Solve $S\Delta\mathbf{y} = -\tilde{\mathbf{g}}$ using BiCGStab.
   - 3.1 Compute $hC\mathbf{w} = \mathbf{x}^{(j)} - \mathbf{x}^c, \quad j \geqslant 1$, where $\mathbf{x}^{(j)}$ is given by (8)
   $$\mathbf{x}^{(j+1)} = \mathbf{F}(\mathbf{x}^{(j)}, \mathbf{y}^c) + \mathbf{f}(\mathbf{x}^c, \mathbf{y}^c + h\mathbf{w}), \quad j = 0, 1, \dots, \quad \mathbf{x}^{(0)} = \mathbf{x}^c.$$
   - 3.2 Compute
   $$S\mathbf{w} = \frac{1}{h}\left(\mathbf{g}(\mathbf{x}^c - hC\mathbf{w}, \mathbf{y}^c + h\mathbf{w}) - \mathbf{g}(\mathbf{x}^c, \mathbf{y}^c)\right).$$
4. Compute $\Delta\mathbf{x} = \mathbf{x}^c - \mathbf{x}^{(j)}, \quad j \geqslant 1$, where $\mathbf{x}^{(j)}$ is given by (9), i.e.
$$\mathbf{x}^{(j+1)} = \mathbf{F}(\mathbf{x}^{(j)}, \mathbf{y}^c + \Delta\mathbf{y}), \quad j = 0, 1, \dots, \quad \mathbf{x}^{(0)} = \mathbf{x}^c.$$

Example 1

Consider

$$0 = x^2 + y^2 - 25$$
$$0 = y - x - 1$$

The Block-Newton method with the usage of all Jacobian converges after 49 iterations. The convergence of the approximated Block-Newton method depends on the choice of h and the number of iteration for the fixedpoint iterations

Example 1

	$j = 3$		$j = 10$	
h	accuracy	steps	accuracy	steps
$h = 1.0e-04$	7.20e-05	361	5.61e+00	401
$h = 1.0e-06$	7.13e-07	401	9.98e-06	401
$h = 1.0e-07$	7.64e-08	401	1.80e-05	401
$h = 1.0e-08$	3.35e-07	401	1.90e+01	401
$h = 1.0e-09$	1.03e-07	353	1.84e-05	401
$h = 1.0e-10$	1.03e-07	229	3.77e-04	401
$h = 1.0e-12$	1.12e-03	401	1.07e+00	401
$h = 1.0e-14$	3.12e-01	401	3.12e-01	401

Example 2

Let $\Omega = (0, 1)$. As a second example we consider the PDE

$$-u'' + 2 + \omega_1(\sin(\omega_2 u)^2 + \cos(\omega_2 v)^2 - 1) = 0$$
$$-v'' + 2 + \omega_3(\sin(\omega_2 u)^2 + \cos(\omega_4 v)^2 - 1) = 0$$

where the exact solution is given by

$$u(x) = v(x) := x^2 + 2x + 1, \quad x \in [0, 1].$$

The Dirichlet condition are computed from the exact solution and the discretisation of the Laplace operator is done with central differences. Let $x_i = ih$, $i = 1, \ldots, N$ with $h = 1/(N + 1)$.

Example 2

Then we get the discretised problem

$$A_h \mathbf{u}_h + \mathbf{f}_h = 0$$
$$A_h \mathbf{v}_h + \mathbf{g}_h = 0,$$

where

$$f_h^{(i)} = h^2 (2 + \omega_1 (\sin(\omega_2 u_h^{(i)})^2 + \cos(\omega_2 v_h^{(i)})^2 - 1)$$
$$g_h^{(i)} = h^2 (2 + \omega_3 (\sin(\omega_4 u_h^{(i)})^2 + \cos(\omega_4 v_h^{(i)})^2 - 1).$$

and $\mathbf{u}_h = (u_h^{(1)}, \ldots, u_h^{(N)})^\top$, $\mathbf{v}_h = (v_h^{(1)}, \ldots, v_h^{(N)})^\top$, $\mathbf{f}_h = (f_h^{(1)}, \ldots, f_h^{(N)})^\top$, and $\mathbf{g}_h = (g_h^{(1)}, \ldots, g_h^{(N)})^\top$. With 10 inner nodes we get the following result for $\omega_1 = \cdots = \omega_4 = 1$:

Example 2

Method	error	CPU-time	it. steps
Fixpointiteration:	1.97e-14	1.30264	757
Newton-scheme:	4.44e-12	0.05010	12
Block-Jacobi:	9.47e-15	0.48683	775
Block-Gauß-Seidel:	9.64e-15	0.51297	811
BAN ($h = 1.0e - 04$, $j = 3$):	3.04e-06	5.63596	100
BAN ($h = 1.0e - 06$, $j = 3$):	3.20e-06	6.78013	100
BAN ($h = 1.0e - 07$, $j = 3$):	3.41e-06	11.97295	100
BAN ($h = 1.0e - 08$, $j = 3$):	6.33e-06	12.85734	100
BAN ($h = 1.0e - 09$, $j = 3$):	7.98e-06	15.10179	100
BAN ($h = 1.0e - 10$, $j = 3$):	6.99e-05	4.78925	100
BAN ($h = 1.0e - 12$, $j = 3$):	2.48e-02	4.80273	100
BAN ($h = 1.0e - 14$, $j = 3$):	7.01e-01	4.43289	100

Example 2

Method	error	CPU-time	it. steps
Block-Newton:	2.03e-15	0.08825	51
BAN ($h = 1.0e-04, j = 10$):	1.38e-07	5.19309	38
BAN ($h = 1.0e-06, j = 10$):	9.79e-08	4.98691	39
BAN ($h = 1.0e-07, j = 10$):	7.23e-07	4.97899	39
BAN ($h = 1.0e-08, j = 10$):	1.61e-06	10.17502	79
BAN ($h = 1.0e-09, j = 10$):	1.09e-05	18.09106	100
BAN ($h = 1.0e-10, j = 10$):	1.21e-04	23.20320	100
BAN ($h = 1.0e-12, j = 10$):	2.37e-02	23.13645	100
BAN ($h = 1.0e-14, j = 10$):	3.52e-01	12.44197	100
BAN ($h = 1.0e-04, j = 20$):	6.23e-08	4.88898	20
BAN ($h = 1.0e-07, j = 20$):	5.21e-07	5.05110	21
BAN ($h = 1.0e-08, j = 20$):	4.24e-06	10.09915	42
BAN ($h = 1.0e-14, j = 20$):	3.78e+00	19.81484	100

Numerical differentiation

One problem is the choice of h (see last example). One might expect that the results are better if a small h is used but unfortunately this is not true. The results become bad if $h \approx 10^{-14}$. The reason is that differentiation is an ill-posed problem. Consider the function

$$f(x) = e^{-x}$$

and assume that $f'(4/10)$ should be computed. We use the formula

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}, \quad h > 0.$$

Numerical differentiation

h	approx	error
1.0e-01	-6.378938632301e-01	3.242618280558e-02
1.0e-02	-6.669795899320e-01	3.340456103646e-03
1.0e-03	-6.699849977048e-01	3.350483308749e-04
1.0e-04	-6.702865311503e-01	3.351488532921e-05
1.0e-05	-6.703166944511e-01	3.351584583311e-06
1.0e-06	-6.703197108493e-01	3.351863409051e-07
1.0e-07	-6.703200128300e-01	3.320567831810e-08
1.0e-08	-6.703200439162e-01	2.119433628600e-09
1.0e-09	-6.703201327340e-01	8.669840834141e-08
1.0e-10	-6.703204658010e-01	4.197653157290e-07
1.0e-11	-6.703193555779e-01	6.904577090072e-07
1.0e-12	-6.703526622687e-01	3.261623302975e-05
1.0e-13	-6.705747068736e-01	2.546608379548e-04
1.0e-14	-6.772360450213e-01	6.915998985706e-03
1.0e-16	-1.110223024625e+00	4.399029785895e-01

Numerical differentiation

One possibility to get better results is to use extrapolation (see Schwarz (2009)). In this case we approximate $f'(x)$ for different h, say for h_i we get an approximation $y_i := f'(x)$. In the second step we compute the interpolation polynomial P which fits the points (h_i, y_i) and finally we compute $P(0)$, i.e. we extrapolate with $h = 0$.

Numerical differentiation

h	approx	error	extra approx	error
1.0e-01	-6.37894e-01	3.24262e-02	-6.37894e-01	3.24262e-02
1.0e-02	-6.66980e-01	3.34046e-03	-6.70211e-01	1.08709e-04
1.0e-03	-6.69985e-01	3.35048e-04	-6.70320e-01	2.73202e-08
1.0e-04	-6.70287e-01	3.35149e-05	-6.70320e-01	7.80154e-13
1.0e-05	-6.70317e-01	3.35158e-06	-6.70320e-01	5.05274e-12
1.0e-06	-6.70320e-01	3.35186e-07	-6.70320e-01	3.02548e-11
1.0e-07	-6.70320e-01	3.32057e-08	-6.70320e-01	3.51979e-10
1.0e-08	-6.70320e-01	2.11943e-09	-6.70320e-01	1.34567e-09
1.0e-09	-6.70320e-01	8.66984e-08	-6.70320e-01	9.76360e-08
1.0e-10	-6.70320e-01	4.19765e-07	-6.70321e-01	4.60815e-07
1.0e-11	-6.70319e-01	6.90458e-07	-6.70319e-01	8.28082e-07
1.0e-12	-6.70353e-01	3.26162e-05	-6.70357e-01	3.67338e-05
1.0e-13	-6.70575e-01	2.54661e-04	-6.70602e-01	2.82060e-04
1.0e-14	-6.77236e-01	6.91600e-03	-6.78059e-01	7.73895e-03
1.0e-16	-1.11022e+00	4.39903e-01	-1.16512e+00	4.94799e-01

Literature

- S. Artlich and W. Mackens. Newton-coupling of fixed point iterations. In G. Wittum W. Hackbusch, editor, Numerical Treatment of Coupled Systems. Vieweg, Braunschweig, 1995.
- Hans Rudolf Schwarz and Norbert Köckler. Numerical mathematics. (Numerische Mathematik.) Studium. Wiesbaden: Vieweg+Teubner, 2009.
- J. Steindorf. *Partitionierte Verfahren für Probleme der Fluid-Struktur Wechselwirkung*. Dissertation, Technische Universität Braunschweig, Braunschweig, 2002.