Surname		Note
First name		
Legi number		
Computer	slabhg	

1	2	3	4	5	Points

- Fill in the cover sheet. (Computer: write the number of the PC as printed on the table). Leave your Legi on the table. Switch off your mobile phone.
- Login to the computer, User name: `student` Password: `ethz`
- You will be asked by the computer for your nethz Account and your Name
- press ALT+F2, then type `gnome-terminal`, then press ENTER.
- Copy the prepared MATLAB files: in console, type `cp resources/matlab/* results` and press ENTER. (Do not execute this during the exam, it will overwrite your results)
- MATLAB: in console, type `matlab&` and press ENTER.
  - MAXIMIZING: move the MATLAB window to the top left corner of the screen, then maximize. Otherwise, the menus might not work properly (a bug!).
  - **Change the current folder in Matlab: in the Command Windows of Matlab, type `cd results` and press ENTER**
  - SHORTCUTS (Ctrl+C, Ctrl+V...): File > Preferences > Keyboard > Shortcuts. Then for "Active settings" choose "Windows default set".
- If you want you can change the keyboard layout: Point with the mouse to the top left corner → write `key` → Keyboard → Layout Setting → + → the desired keyboard layout→ Add. Choose the desired Keyboard layout at the top right corner of the Computer screen. **(Only before the exam starts!!!)**
- Don't use pencils, red or green pens.
- Wait the signal from the assistant before to look at the exam sheet.

- Write your name on every page.
- At the end of the exam, **only when you are told by the assistant**,
  - in console, type `cd resources/` to move into the folder `~/resources/`
  - in console, type `./makepdf.sh` (only once);
  - when requested insert name, surname and Leginummer (avoid Umlaute, accented characters and underscores)
  - a PDF file will appear, check if it contains all your Matlab `.m` files and `.eps` figures.
  - if it contains all the files you have to hand in, click on "print" (or press CTRL+P).
  - an assistant will bring you the printout of the PDF; check it again, then hand in.
  - LEAVE YOUR COMPUTER ON, DO NOT TURN IT OFF.

Good luck!

P. Arbenz
S. Pauli

Fall semester 2012

Numerical Methods for CSE

ETH Zürich
D-INF

Examination

January 29th, 2013

Duration of examination: 180 minutes. **Total points: 180**

Problem 1 Sparse matrix

In this problem we use the sparse matrix

```
A = delsq(numgrid('C',n));
```

(1a) [10 points] Write a Matlab script

```
function fillin()
```

which measures the operator complexity (sum of the nonzeros of all factors of a given factorization divided by the number of nonzeros of A) of different decompositions, and plots them for $n = 2^1, \dots, 2^8$. Measure the operator complexity of the following things:

- Cholesky decomposition of `A`
- Incomplete Cholesky decomposition of `A` (see `ichol`)
- propose and implement an alternative way to reduce the fill in of the Cholesky decomposition.

Is is possible to measure an operator complexity number below 1? Explain why.

HINT: You may use the Matlab function `nnz`, and `ichol`.

(1b) [15 points] Plot the runtime (time to compute the solution $\mathbf{x}$, eg. factorization and backward/forward substitution) and the measured error (norm of the residual) for $n = 2^1, \dots, 2^8$ of the three different approaches (of (1a)) to solve $\mathbf{A}\mathbf{x} = \mathbf{b}$. Use a random vector $\mathbf{b}$. Implement this functionality in the Matlab script

```
function conf()
```

Comment on the obtained results.

HINT: You may use `tic`, `toc`, `rand`, `norm` and `sort`.

HINT: Read (1c) before starting this sub problem

HINT: As you measure the runtime call `maxNumCompThreads(1)` to insure that Matlab is running in single core mode.

(1c) [10 points] Propose and implement (in (1b)) an alternative method which takes advantage of the sparsity of $\mathbf{A}$, and solves $\mathbf{A}\mathbf{x} = \mathbf{b}$ not exactly but with a residual norm of 10^{-3}.

Comment on the obtained results (may be together with (1b)).

Problem 2 Linear least squares

Let two vectors $\mathbf{z}, \mathbf{c} \in \mathbb{R}^n$, $n \in \mathbb{N}$ of measured data be given. The two numbers α^* and β^* are defined as

$$(\alpha^*, \beta^*) = \underset{\alpha,\beta\in\mathbb{R}}{\operatorname{argmin}} \|\mathbf{T}_{\alpha,\beta}\mathbf{z} - \mathbf{c}\|_2, \tag{1}$$

with the tridiagonal matrix

$$\mathbf{T}_{\alpha,\beta} = \begin{pmatrix} \alpha & \beta & 0 & \dots & 0 \\ \beta & \alpha & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \alpha & \beta \\ 0 & \dots & 0 & \beta & \alpha \end{pmatrix} \in \mathbb{R}^{n,n}.$$

(2a) [8 points] Reformulate (1) as a linear least squares problem in the usual form

$$\mathbf{x}^* = \underset{\mathbf{x}\in\mathbb{R}^k}{\operatorname{argmin}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2$$

with suitable $\mathbf{A} \in \mathbb{R}^{m,k}$, $\mathbf{b} \in \mathbb{R}^m$, $m, k \in \mathbb{N}$.

(2b) [13 points] Write a Matlab function

```
[alpha, beta] = lsqest(z,c)
```

that computes the values of the optimal parameter α^* and β^* according to (1) from the data vectors `z` and `c` (i.e., $\mathbf{z}$ and $\mathbf{c}$). Use the QR-decomposition to solve the linear least squares problem.

HINT: For $\mathbf{z} = (1, 2, \dots, 10)^T$ and $\mathbf{c} = (10, 9, \dots, 1)^T$ you should get $\alpha^* \approx -0.4211$ and $\beta^* \approx 0.5789$.

Problem 3 Bézier semi-circle

(3a) [12 points] Write a Matlab function

```
function  plot_bezcurv(d)
```

which draws the Bézier curve, the control points $\mathbf{d}$, and the convex hull defined by the control points $\mathbf{d}$ ($2 \times n$ matrix).

HINT: You may use `convhull` and `fill`

(3b) [12 points] Write a Matlab function

```
len = function  bezLength(d)
```

which approximately computes the length of the Bézier curve by some numerical quadrature.

(3c) [16 points] We want to approximate a unit half circle by (one segment of) a Bézier curve with 5 Bézier control points. The Bézier control points shall be such that

- the Bézier curve is symmetric.
- the Bézier curve passes through the points $\binom{-1}{0}$, $\binom{0}{1}$ and $\binom{1}{0}$.
- the tangent at the two end points $\binom{-1}{0}$ and $\binom{1}{0}$ corresponds to the tangent of the semi circle.
- the length of the Bézier curve corresponds to the length of a semi circle (π).

Write a Matlab function

```
function  halfcircle(d)
```

where you compute the Bézier control points and plot the resulting Bézier curve.

HINT: You may use the following 5 Bézier control points: $\binom{-1}{0}$, $\binom{-1}{\alpha}$, $\binom{0}{\beta}$, $\binom{1}{\alpha}$ and $\binom{1}{0}$

HINT: Find a linear (analytical) relation between α and β such that the Bézier curve passes through the point $\binom{0}{1}$. Then only one independent parameter is left. Then use a **simple** numerical method from the lecture to determine a good value for this parameter such that the Bézier curve has length π

HINT: If you are unable to complete task (3a) and/or (3b) use the functions `plot_bezcurv_p(d)` and/or `len = bezLength_p(d)` instead.

Problem 4 Best rank-1 approximation

Given $\mathbf{A} \in \mathbb{R}^{n,n}$ with positive diagonal we consider the minimization problem

$$\mathbf{x}^* = \underset{\mathbf{x}\in\mathbb{R}^n}{\operatorname{argmin}} \left\|\mathbf{A} - \mathbf{x}\mathbf{x}^\top\right\|_F^2 . \tag{2}$$

(4a) [6 points] Reformulate (2) as a standard non-linear least squares problem $\frac{1}{2}\left\|F(\mathbf{x})\right\|_2^2 \to \min$ for a suitable function F.

(4b) [15 points] Derive the Jacobian and implement it in the Matlab function

```
function df = DF(x).
```

(4c) [11 points] Write a Matlab code

```
function x = rankoneapprox(A)
```

that computes the solution of (2) by means of the Gauss-Newton iteration with initial guess $x_i^{(0)} = \sqrt{a_{ii}}, i = 1, \dots, n$ with a tolerance of 10^{-3}.

HINT: You may use `reshape`.

HINT: If you are unable to complete task 2 use the function `df = DF_p(x)` instead.

HINT: Use `test4.m` to test your function.

(4d) [8 points] Explain, why the MATLAB built in function `eig` can be used to solve (2) provided that $\mathbf{A}$ is *symmetric* (hermitian).

(4e) [5 points] Write a MATLAB code

```
function x = symrankoneapprox(A)
```

that computes the solution of (2) for *symmetric* (hermitian) $\mathbf{A}$ using MATLAB's `eig`.

HINT: Use `test4.m` to test your function.

Problem 5 ODE / Runge-Kutta method

We want to solve the initial value problem $\dot{\mathbf{y}} = \mathbf{f}(t, \mathbf{y})$, $\mathbf{y}(t_0) = \mathbf{y}_0$ by the Runge-Kutta method characterized by the following Butcher table:

$$\begin{array}{c|ccc} 0 & 0 & & \\ \frac{2}{3} & \frac{2}{3} & 0 & \\ 0 & -1 & 1 & 0 \\ \hline & 0 & \frac{3}{4} & \frac{1}{4} \end{array} \tag{3}$$

(5a) [11 points]

Implement two Matlab functions

```
y1 = RK_step(odefun, t, y0, h),
```

and

```
[t, y] = RK(odefun, tspan, y0, N).
```

`RK_step(odefun, t, y0, h)` implements one Runge-Kutta step defined by (3), from `t` to `t+h` with the starting value `y0`. `odefun(t,y)` defines the right hand side of the initial value problem $\dot{\mathbf{y}} = \mathbf{f}(t, \mathbf{y})$, $\mathbf{y}(t_0) = \mathbf{y}_0$.

`RK(odefun,tspan,y0,N)` uses `RK_step` to solve the ODE over the whole interval specified by `tspan` using $N \in \mathbb{N}$ uniform timesteps.

(5b) [5 points] Bring the following ODE into an suitable form to solve it with `RK_step(odefun, t, y0, h)`:

$$x''(t) + x(t) = \sin(t), \quad x(0) = 100, \quad x'(0) = 5. \tag{4}$$

Implement this suitable form in the Matlab function

```
out = odefun(t, y).
```

(5c) [12 points] Implement a Matlab function

```
exRK(),
```

to graphically (with a plot) determine the order of the chosen Runge-Kutta method (3) for the given ODE (4). Use the Matlab function `ode45` with relative tolerance `100 * eps` and absolute tolerance `eps` to determine a reference solution.

HINT: The error of a method could be computed by $\|x(T) - \hat{x}(T)\|_{L_2} + \|x'(T) - \hat{x}'(T)\|_{L_2}$, where x is the reference solution and $\hat{x}$ the approximate one. T is the stopping time.

HINT: You may use `norm`.

HINT: If you are unable to complete task (5a) and/or (5b) use the functions `[t, y] = RK_p(odefun,tspan,y0,N)` and/or `out = odefun_p(t, y)` instead.

(5d) [13 points] Analytically determine the convergence order and the stability interval of the Runge-Kutta method in (3)