Karlsruhe Institute of Technology (KIT) ST 2013
Institute for Applied and Numerical Mathematics – RG 4
Numerical Simulation, Optimization and HPC

PD Dr. Gudrun Thäter
Dipl.-Math. Leonid Chaichenets
Dipl.-Math. techn. Eva Ketelaer

Project-oriented Software Lab

Project 2 — Convection-Diffusion Equation

<u>Presentation due on:</u> Fri. July 12 2013

Exercise 1:
Consider the following stationary convection-diffusion equation:

$$\begin{aligned} a \cdot \nabla u - \nabla \cdot (\nu \nabla u) &= s && \text{in } \Omega, \\ u &= u_D && \text{on } \Gamma \subseteq \partial\Omega, \end{aligned}$$

where u is the scalar unknown function, e.g. the concentration of a solute, a is the given convection velocity (also known as advection), ν is the coefficient of diffusivity and s is a volumetric source term. u_D denotes the prescribed values at the Dirichlet boundary Γ.

Derive the weak form for this boundary value problem!

Exercise 2:
Implement the local assembly (weak form) for the convection-diffusion equation in the code skeleton provided. Use $s \equiv 0$ as the right-hand side, $\nu = 0.1$, a and the Dirichlet conditions as indicated in Figure 1. Note that there is no boundary condition set at the right and the top of the unit square.

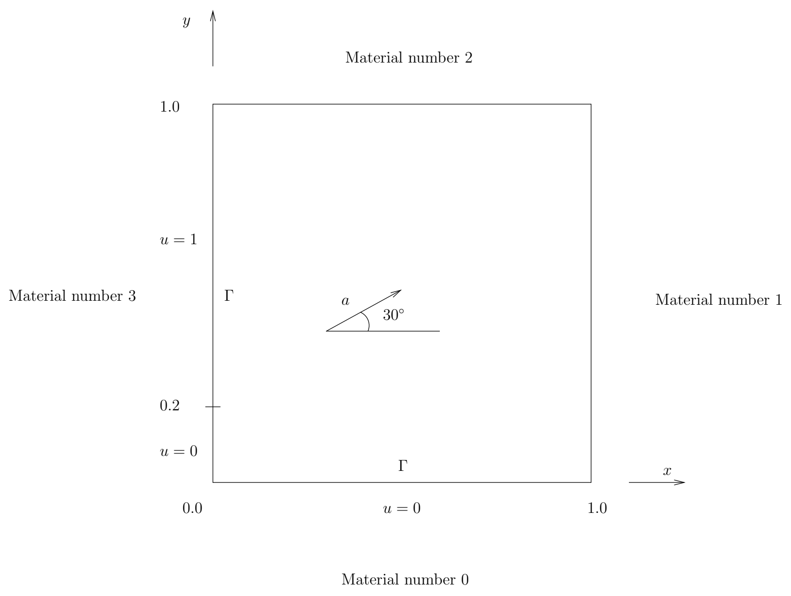


Figure 1: Setup for the convection-diffusion problem in the unit square. $\|a\| = 1$, constant.

-Please turn over!-

Exercise 3:
The Galerkin finite element method that we usually use in HiFlow^3 is not perfectly suited to solve convection-dominated problems. The mesh Péclet number

$$Pe = \frac{\|a\|\, h}{2\nu},$$

with the mesh parameter h, characterizes the relative influence of convection and diffusion in a given flow problem.

(a) Investigate the value of the Péclet number with respect to the stability of the solution! Modify it by changing the value of the diffusivity ν and the refinement level. What is the maximum value of Pe yielding a stable system?

(b) Usually the diffusivity can not be chosen freely, but is fixed due to the underlying physical problem. Implement the streamline-upwind Petrov-Galerkin method (SUPG) to stabilize the system! What effect does this have on the constraints on Pe?

(c) An issue in the SUPG method is the added stabilization term, which is not symmetric, thus makes it difficult to establish statements on the stability. This can be avoided with a Galerkin/Least-squares (GLS) technique. Implement a GLS stabilization and compare SUPG to GLS with respect to implementation difficulties, efficiency and quality of the solution!

Exercise 4:
Consider the following time-dependent convection-diffusion equation:

$$\begin{aligned} u_t + a \cdot \nabla u - \nabla \cdot (\nu \nabla u) &= s \text{ in } \Omega \times (0, T], \\ u(\cdot, 0) &= 0 \text{ in } \Omega, \\ u &= 0 \text{ on } \partial\Omega \times [0, T], \end{aligned}$$

where

$$\begin{aligned} a &= \sqrt{(x - \frac{1}{2})^2 + (y - \frac{1}{2})^2} * (-(y - \frac{1}{2}), x - \frac{1}{2}), \qquad \nu = 10^{-5}, \\ s &= e^{-t^{10}} * \begin{cases} \cos(2\pi * \sqrt{(x - \frac{1}{4})^2 + (y - \frac{1}{4})^2}), & \text{if } \sqrt{(x - \frac{1}{4})^2 + (y - \frac{1}{4})^2} \leq \frac{1}{4} \\ 0, & \text{else.} \end{cases} \end{aligned}$$

We want to solve this equation with the method of lines, which means that first the spatial derivatives are discretized while the time derivative is left continuous. This leads to a system of ODEs in time that can be solved using time discretization methods.

(a) Implement the θ family of methods:

$$\frac{u(t^{n+1}) - u(t^n)}{\Delta t} = \theta u_t(t^{n+1}) + (1 - \theta) u_t(t^n) + \mathcal{O}((\frac{1}{2} - \theta)\Delta t, \Delta t^2)$$

(b) Implement either a fractional-step method (e.g. fractional-step-θ scheme) or a higher-order time-stepping scheme (e.g. fourth order Runge-Kutta method).

(c) Derive the local and global truncation error for one of the implemented schemes.

(d) What are the stability domains of the different schemes? Experiment with the size of the timestep Δt! Investigate how this is related to the stability domain!

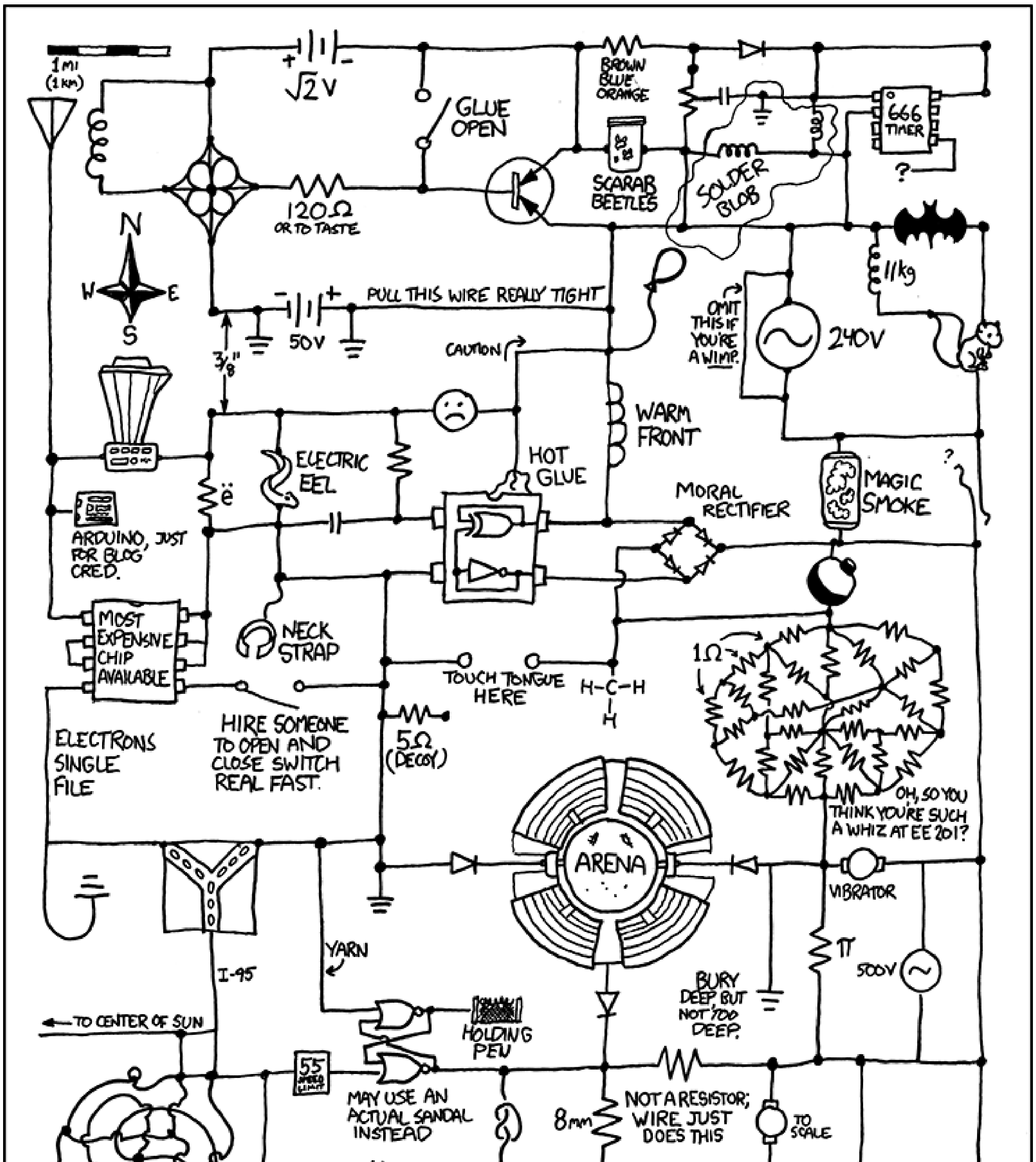
Exercise 5:
In the last 50 years, Moore's law described the long-term trend of transistors that fit on an integrated circuit. This meant that every two years, the speed of CPUs and therefore of the algorithms doubled automatically. Due to electron density and energy issues, this paradigm is changing and parallelism plays an important role by now. The problem is that unlike doubling of the performance, doubling of the cores does not mean automatic increase of the speed of algorithms. Even worse, algorithms need to be adapted to run on parallel platforms.

(a) What is "speedup" and "efficiency"? Find out about laws on parallel performance!

(b) Log in on the provided multicore system, compile your code on that platform and run speedup tests with 1, 2, 4 and 8 processes. Interprete your results.

(c) *Optional:* Repeat the previous exercise for the code of your first project.

-Please turn over!-



Source: http://www.xkcd.com/730/ **Copyright:** Randall Munroe