Version 10/11 2016,
François Lauze,
revised from Jens Hugger's 2/11/2015 version.

NumDiff: Numerical Solution of Differential Equations
Block 3A, 2017

What's happening in NumDiff?

NumDiff is a course of the type that makes Copenhagen University [KU] what it is - **strong on theoretical foundation**.
KU graduates not only are able to apply the newest methods – they also understand what lies behind so that they can participate in creating the methods of the future.

We do see some "real life" applications through examples and weekly assignments, but the course focus is on the theoretical content of the methods - **comparing and evaluating various methods for the same problems with respect to for example cost and precision**.

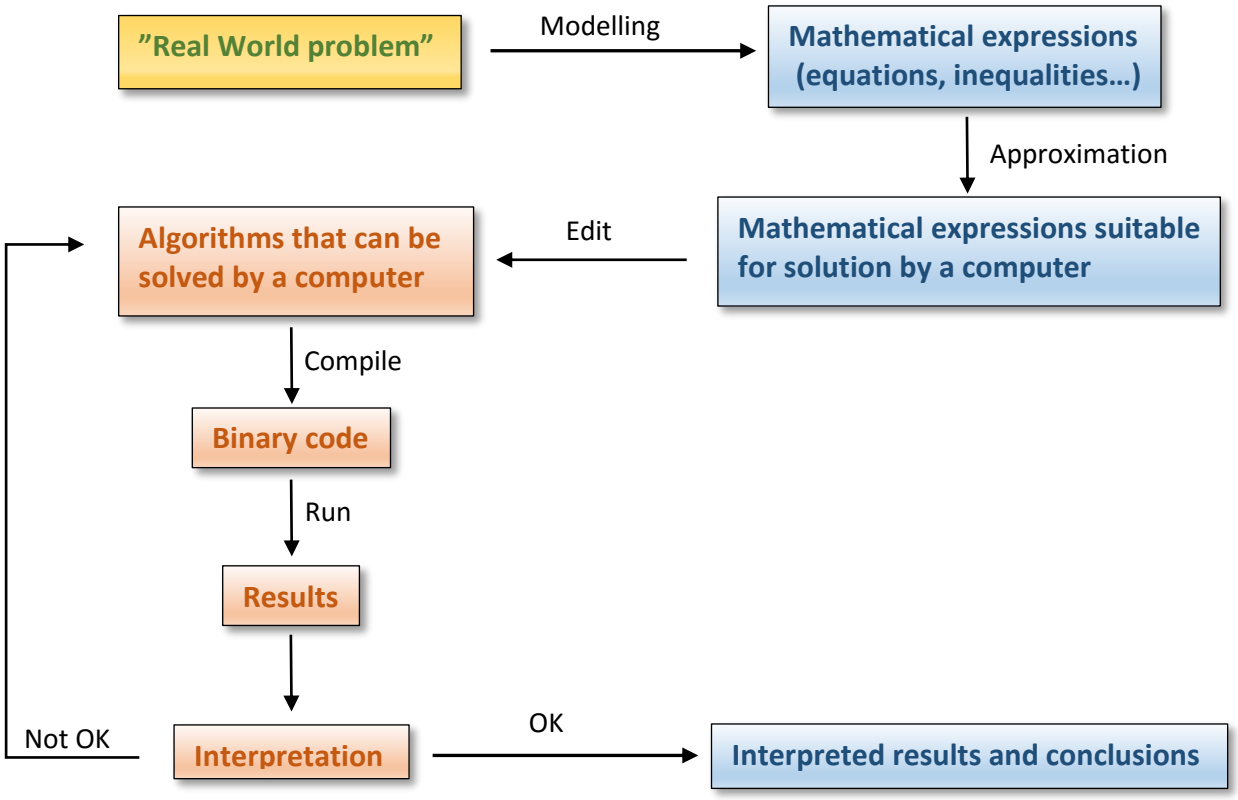
NumDiff contains the **classical numerical (Finite Difference) methods for the solution of differential equations in both one and more dimensions (ordinary and partial differential equations)** only briefly touching on the more advanced integral methods (Collocation and Finite Element). You learn the notions and the methods that all else is built on top of.
These are methods in practical use today but more often they are the foundations of the state of the art methods being used and researched today.
In NumDiff we also try out the methods **- writing and running code**. We focus on "toy-problems" small enough that it is possible to focus on the methods and not get caught in "size-problems". Follow up courses to NumDiff could be named "NumDiff - Size matters" or "NumDiff – Integral methods".

Why is NumDiff important?

You learn programming in practice.
You learn the basic foundation of the numerical solution of dynamic problems.
You learn how to take the step from "solving exercise problems" to "writing a small project".
Applications:

1. **During your studies**: Within insurance, finance, operations research, physics, chemistry, geology, biology, and you name it - you often need to solve your models on a computer. Whenever the problems are dynamic (changes with time or any other parameter) this will, one way or the other, involve the methods learned in this course.
2. **After your studies**:
    a. If you become a researcher you will need to solve your own dynamic models or help others solve theirs.
    b. If you become a high school teacher you will need the programming and numerical skills in interdisciplinary projects.
    c. If you go to industry, you will need a training period. In this period, you will typically "earn your salary" as a programming help for a project group.
    d. The ability to understand and numerically solve differential equations is required in many fields: as a quant in the financial world, as a planner in the Operations Research world and so on.

What is Numerical Analysis?



Focus in NumDiff:

1. **A little bit of "modeling"**.
   The textbook contains examples taken from real life and the weekly assignments are originating in problems taken from real life.
2. **A lot of "Approximation"** in the cases where the mathematical expressions involve differential equations and large linear equation systems.
3. **Quite a few "Edit-Compile-Run-Error correct"-cycles** on expressions of the types above. Python is mainly used during lectures (though Maple might be) and supported for exercises, but there is freedom of choice for handins.
4. **Quite some "Interpretation" of results**, in the weekly assignments.

Lecturers:

- François Lauze (FL, contact person)
- Knud Henriksen (KH)

Teaching Assistant(s) (TA's) :

- To be announced.

Course contents:

The goal is that you learn some basic tools needed for solving mathematical problems on a computer:

1. Learn the necessary mathematics for solving differential equations in one or more dimensions on a computer using the "Finite Difference Method".
2. Extend your programming skills on how to make a computer run the mathematical algorithms that you learn to construct. The programming language used in lectures is Python and there are lectures on programming in Python. Also, students with no prior Python skills are helped by a repetition /access to the Python programming lectures from NumIntro. The TA's will be able to help with programming problems in Python. You are however allowed – at your own responsibility – to use your favorite

programming language for weekly assignments, but programming language specific help is only guaranteed for Python and you will meet a substantial amount of Python code in the lectures and exercises.

There are the following acknowledged problems in using Python instead of Maple for the weekly assignments:

- Week 1 problem 1.1.c: You are asked to do a Taylor expansion in Maple. It should be possible to do this with the background knowledge from MatIntro and LinAlg and the help in the assignment itself. The **`sympy`** (Symbolic Python) Python module can be used to implement Taylor series. If not, and if your programming language is not able to do symbolic Taylor expansions, you may do the Taylor expansion by hand (always a good idea).
- Week 4 problem 4.2d+e: You are asked to solve a problem using different numbers of computational digits. Python has a package **`bigfloat`** allowing for this, but as of now (November 2016) we have little experience with this package (this should hopefully change).
- Week 8-9 problem 4: You are asked to solve a nonlinear system of second order PDE's with the built-in Maple **`PDEtools`** package. This may be replaced by another package of your choice if using another programming language. For Python, there are several packages for solving PDE's, but as of now (November 2016) we have no experience with these (but this should hopefully change too).

Course topics:

Math-topics:

1. Background stuff from **mathematics.**
2. Finite difference methods for **initial value problems for ordinary differential equations.**
3. Finite difference methods for **boundary value problems for ordinary differential equations.**
4. Finite difference methods for **diffusion problems for partial differential equations.**
5. Finite difference methods for **advection problems for partial differential equations.**
6. Finite difference methods for **wave problems for partial differential equations.**
7. Finite difference methods for **elliptic problems for partial differential equations.**

Python topics for students with no or little prior knowledge about programming in Python:

1. **Basic Python.**
2. **The Spyder IDE (Scientific PYthon Development EnviRonment)**
3. **Flow control, Functions.**
4. **Numerical `Programming` with focus on `numpy` and `scipy` modules.**
5. **Plotting with** matplotlib**.**
6. **Parallel programming, potential use of `bohrium` module?**
7. **Classes and object oriented programming.**

Literature:

Compulsory literature (Text book):

- Mark H. Holmes, "Introduction to Numerical Methods in Differential Equations", 2007, Springer, Texts in Applied Mathematics TAM-52, ISBN 0-387-30891-1. (Buy it). Note the accompanying webpage. (See folder "Course material").
- Jens Hugger, "Supplemental notes to Holmes App. A – Taylor and rounding errors. (Free on Absalon).
- Jens Hugger, "Supplemental notes to Holmes pp. 7-13 (with references to pp. 31-32 and 118)". (Free on Absalon).
- Lecture notes supplementing but not replacing the book. You have to study the book, listen to the lectures and read the lecture notes. If you have problems with the book maybe the answers are in lecture notes. (Free on Absalon. See folder "Course material"):
    - Jens Hugger, Maple theory (NumIntro).
    - François Lauze, NumDiff lecture notes.
    - Knud Henriksen, NumDiff lecture notes.

About Python programming:

The amount of material freely available on the World Wide Web is absolutely huge. A good start for a list of textbooks and tutorials is at the main Python website. A lot of them are freely available. Documentation on **`numpy`** and **`scipy`** is available at www.scipy.org. Documentation on **`matplotlib`** is available at its main website. In general, StackOverflow is an invaluable resource (and not just for Python).

Supplementary reading:

For ch. 1-2:

- Jens Hugger, Numerical solution of DEP, 2013. (See folder "Course material").

On PDE's:

- PDE survey on ScholarPedia
- J. W. Thomas, "Numerical Partial Differential Equations: Finite Difference Methods", Springer TAM 22.

On Python:

- Again, so much documentation is available that it may feel a bit overwhelming. However, the material from J. R. Johansson on GitHub seems a good start, look first at the online read-only version.

On Maple:

- Maple, User guide. Maple, Programming guide. (Free on KUnet->Softwarebiblioteket->Maple).
- Maple, Programming guide. (Free on KUnet->Softwarebiblioteket->Maple)

Confrontation and home work:

Listening, theory training and reading:
Each week starts with a lecture Tuesday 8-11 rounding off the previous week and commencing the current week. The theory of the current week is finished with a second lecture Thursday 10-12**. Attend these lectures and read the book and eventually the lecture notes!**
Week 1-4 hold additionally each one double lecture focused on Python. **Attend these and read the lecture notes and/or the Python manuals if you are not a fluent programmer (or followed the similar lectures in NumIntro)!**
Theory is not enough to learn the subjects. You must also experiment:
Each week holds 5 hours of exercise sections where you can get help from a teaching assistant. For each week, there is a compulsory weekly assignment. **The compulsory assignments must be handed in for grading by the instructor.**

Weekly assignments and final exam:

- **7 weekly compulsory assignments**.
  The weekly assignments should be worked out and handed in by groups of 3 no later than midnight, Sunday of the week, through Absalon, in the form of a runnable Python source code (or the source code of the programming language of your choice) with all results visible on a pdf-file. For the 7th assignment, the deadline is by midnight, Friday of the week.
  Individual hand in's or groups of 2 or 4 is by permission only.
  To be able to participate to the final exam, it is required that the 7 weekly assignments are approved and valid Monday in week 8 of the block. (An assignment expires after Block 3 of the following year).
  If a weekly assignment is failed, it can be resubmitted once in a corrected version one week after the original hand in date through Absalon, in the form of a runnable Python source code (or the source code of the programming language of your choice) with all results visible on a pdf-file, but with file names different from the original submission (to not overwrite the original submission). The 6th weekly assignment must be resubmitted by midnight Friday of week 7 in block 3. The 7th and last weekly assignment can be resubmitted only as part of the reexam. (But see "fast reexam" below).
  A week 1-6 assignment is approved when all problems are solved correctly with minor mistakes and omissions (corresponding to a grade of at least 7).
  A week 7 assignment is approved if it is graded to at least 2 (the lowest passing grade).
- The final exam consists of a final project to be worked out in week 8-9 and handed in individually no later than midnight Sunday at the end of Block 3. The course is pass/fail based on the final exam except for exchange students that can demonstrate the need for a grade (see below).

- Reexam: Same as the final exam except that only one week (full time) is allowed for the final project that must be turned in no later than midnight Sunday at the end of Week 26. Those of the 7 weekly compulsory assignments that are not approved must be handed in no later than midnight Sunday at the end of Week 25.
- Fast reexam in block 3: If the first 6 weekly assignments are approved by Monday in week 8 of block 3 but the 7th weekly assignment is not approved by this deadline but is resubmitted no later than midnight Sunday of week 8 of block 3 – and passes, then the final project will be accepted and graded together with the final projects for the ordinary exam.

A grade for an exchange student is given based on the final project.
Exchange students requiring a grade must ask the teachers to get a grade no later than in the 3rd week of the course.

Weekly plan for NumDiff, Block 3A, 2017, weeks 1-6

Mon	Tuesday		Wed	Thursday		Fri	Sat	Sunday
	8:15--11:00	Lecture 1 Loc. TBA		8:15-10:00	Lecture 2 Loc. TBA			Hand-in week's homework before midnight. Resubmit if needed last weeks homeworks before midnight.
	11:15-12:00	Weekly assignments with consultation Bring laptop. Loc. TBA		10:15-12:00	Python programming. Loc TBA			
		Study lec1 (after) + Study lec2 (before)		12:15-13:00				
				13:15-17:00	Weekly assignment with consultation. Bring Laptop. Loc TBA			
					Finish weekly assignment			

Weekly plan for NumDiff, Block 3A, 2017, week 7.

Mon	Tuesday		Wed	Thursday		Fri	Sat	Sun
	8:15--12:00	Lecture 1+2 Loc. TBA		8:15-10:00	Weekly assignment with consultation. Bring Laptop. Loc TBA			
				10:15-12:00	Weekly assignment with consultation. Bring Laptop. Loc TBA			
		Study lec1+2 (after)		12:15-13:00				
				13:15-15:00	Weekly assignment with consultation. Bring Laptop. Loc TBA			
				15:15-17:00	Background for final exam. Loc. TBA			
					Finish weekly assignment Hand in this week's homework before midnight. Resubmit if needed last weeks homeworks before midnight.			

Study Program – NumDiff 2017: Lectures and exercises

Week	Time	Name	Topic/Problems
6[1] 7+9/2	Tu 8^{15}-9	Course Intro [all] Lec. 0 [FL]	**Intro lecture** a) Presentation of the teaching team b) NumDiff Lec. 0
	Tu 9-11	Lec. 1 [FL]	a) Order symbol "Big Oh" b) Taylor Expansion c) Round-off error, the smile curve.
	Tu 11-12	Week 1.1 TA	**Week 1 topics and assignments.** Weekly assignment
	Th 8^{15}-10	Lec. 2 [FL]	Holmes, P 1-8, a) Differential equations – concepts and wellposedness b) *y'=Ay* and matrix exponential. c) 1-4 of the 5 steps for constructing a numerical algorithm: Difference d) formulas. (Truncation error and consistency explained in Lec 3).
	Th 10-12	Python [TA, FL]	Basic Python, Flow control, Spyder
	Th 13-17	Week 1.2 [TA]	Weekly assignment.
7[2] 14+16/2	Tu 8^{15}-11^{30}	Lec. 3 [FL]	**Holmes Chap. 1 P-8-18, A-stability and matrix exponential.** a) Discretization and round-off error. Error measures. b) Truncation term, truncation error and local truncation error. c) Convergence: Handout for "Lax: A consistent method that is 0-stable. (contd. Dependence on data) is convergent". d) A-stability as "behavior for finite step sizes". e) Examples of methods (table 1.3).
	Tu 11^{30}-12	Week 2.1 TA	Week 2 topics and assignment Weekly assignment.
	Th 8^{15}-10	Lec. 4 [FL]	**Holmes Pp 18-33** a) Quadrature and Runge-Kutta-methods as examples of table 1.3. b) Extensions and ghost points: Better understanding of step 3 and order of consistency depends also on the order of the boundary condition approx. c) Conservative methods: "Selecting methods not just for order and stability but to mimic special properties".
	Th 10-12	Python [TA, FL]	Functions, **`numpy, matplotlib`**
	Th 13-17	Week 2.2 [TA]	Weekly assignment
8[3] 21+23/2	8^{15}-10	Lec. 5 [KH]	**Holmes Pp.45-58 (Ch.2)** a) Examples of methods. b) Losing the arrow of time. c) Repeat: Step 1-4 in constructing an FDM. d) Tridiagonal matrices. e) Modelling and roundoff errors.

			f) Repeat: More complicated boundary conditions. g) Nonlinear problems – Newtons method.
	Tu 10-11	Week 3.1 [TA]	**Week 3 topics and assignments** Weekly assignment
	Tu 11-12	Week 3.1 [FL, KH]	Week 3 course status
	Th 8^{15}-10	Lec. 6 [KH]	**Holmes Pp.58-73 (Ch.2)** a) Residual methods b) Homogenization c) Shooting methods – regaining arrow of time, transforming higher order ODEs to system of first order ODEs. d) Solution of linear, homogeneous difference equations (for prob. 2.27)
	Th 10-12	Python [TA, FL]	More **`numpy, matplotlib`**
	Th 13-17	Week 3.2 [TA]	Weekly assignment
9[4] 28/2 + 2/3	Tu 8^{15} -11	Lec. 7 [KH]	**Holmes Pp.83-100 (Ch.3) – Ch 1-2 survey Ch 3-6 and Lec 7-14.pdf pp. 3.1-3.4** a) How to construct an FDM (the 5 steps) b) Stability and the connection to convergence. c) 2^{nd} order, linear 2D PDE's and their classification. d) Properties of the exact solution of the heat equation. e) Step 1-4 for the explicit FDM for the heat equation. f) Error analysis for the explicit FDM for the heat equation.
	Tu 11-12	Week 4.1 [TA]	**Week 4 topics and assignments** Weekly assignment
	Th 8^{15}-10	Lec. 8 [KH]	**Holmes Pp.100-119 (Ch.3) – Lec 7-14.pdf pp. 3.5-3.9** a) Implicit method and matrix formulation. b) Theta method and stability. c) Crank-Nicolson method. d) L-stability. e) Dimension reduction (MOL) and Collocation.
	Th 10-12	Python [TA, FL]	Classes, input/output, **`scipy`**
	Th 13-17	Week 4.2 [TA]	Weekly assignments
10[5] 7+9/3	Tu 8^{15} -10	Lec. 9 [FL]	**Holmes Pp.119 (Ch.3) – Lec. 8 extra - Burger's eqn** f) Nonlinear equations (Burger) in particular CN for nonlinear equations. **Holmes Pp.127-132 (Ch.4) – Lec 7-14.pdf Pp. 4.1-4.3** a) Properties of the exact solution to the advection equation. b) Boundary conditions. c) Weak solutions. d) Examples of methods for advection. e) Truncation error for advection methods.
	Tu 10-11	Week 5.1 [FL]	**Background lecture for weekly assignment for week 4-5:** François Lauze, Linear and nonlinear elliptic PDE's - background for Euler- Lagrange eqns. in computer imaging.
	Tu 11-12	Week 5.2 [TA]	**Week 5 topics and assignments** Weekly assignment
	Th 8^{15} -10	Lec. 10 [FL]	**Holmes pp.132-147 (Ch.4) – Lec 7-14.pdf pp. 4.4-4.7**

			a) Stability for advection methods. b) CFL and monotonicity. **Ch 4 survey Ch 5 preview**
	Th 10-12	Python [TA, FL]	Classes, 3D plotting with **`matplotlib`**.
	Th 13-17	Week 5.3 [TA]	Weekly assignment
11[6] 14+16/3	Tu 8^{15} -10	Lec. 11 [KH]	**Holmes Pp.155-164 (Ch.5) – Lec 7-14.pdf Pp. 5.1-4.5** a) Properties of the exact solution to the advection equation.
	Tu 10-11	Week 6 [RP?]	**Background lecture for weekly assignments for week 6:** Rolf Poulsen, Solving financial derivatives
	Tu 11-12	Week 6.1 [TA]	**Week 6 topics and assignments** Weekly assignment
	Th 8^{15} -10	Lec. 12 [KH]	**Holmes pp.164-176 (Ch.5) – Lec 7-14.pdf pp. 5.6-5.8** b) Examples of methods for waves. c) Stability for wave methods. d) Practical behavior of wave methods. **Ch 5 survey Ch 6 preview**
	Th 10-12	Python [TA, FL]	Object oriented programming
	Th 13-17	Week 6.2 [TA]	Weekly assignment
13[7] 28+30/3	Tu 8^{15} -10	Lec. 13 [KH]	**Holmes Pp.181-192 (Ch.6) – Lec 7-14.pdf Pp. 6.1-6.10** **Week 7 topics and assignments** a) Properties of the exact solution to the Laplace equation. b) A method for Laplace's equation. c) The Laplace matrix.
	Tu 10-12	Lec.13 cont. [KH] with some pieces removed?	**Holmes pp.192-214 (Ch.6) – Lec 7-14.pdf pp. 6.5 - 6.10 – Week 7 topics and assignment** a) Properties of the Laplace matrix. (Holmes § 6.2.2 from TEST 1). b) Iterative methods for solving the Laplace equation system – Descent methods. ¨ c) Steepest descent method. d) Conjugate gradient methods. e) Preconditioned Conjugate Gradient methods (Holmes 6.5, maybe).
	Th 8-10	Week 7.1 [TA]	Weekly assignment
	Th 10-12	Python [TA, FL]	Parallel programming, **`bohrium`**?
	Th 13-15	Week 7.2 [TA]	Weekly assignment
	Th 15-16	Week 8-9 [FL]	**Background lecture for final project for week 8-9:** Crank-Nicolson for nonlinear problems.
	Th 16-17	Week 8-9 [PGS?]	**Background lecture for final project for week 8-9:** Preben Graa Sørensen, Patterns on snail shells.
14[8]	-	-	Questions by email or appointment
15[9]	-	-	Questions by email or appointment