Numerical Methods I

MATH-GA 2010.001/CSCI-GA 2420.001

Benjamin Peherstorfer
Courant Institute, NYU

Based on slides by G. Stadler and A. Donev

Outline

Organization

Conditioning of problems

Stability of algorithms

Representing real numbers

Organization

- Time and location: Mondays 5:10–7:00PM, WWH 101
- Office hours: Thursdays, 5-6pm, stop by or make an appointment (please email). My office number is WWH #421; zoom link for online meeting https://nyu.zoom.us/j/99971119262
- Course webpage: https://docs.google.com/document/d/1UzteBLTrHdUG42sW4AQg6pUIM5qgoceVABnknpTvX0Y/edit?usp=sharing

  You need to be logged in with your NYU-Google account to access it
- Brightspace https://brightspace.nyu.edu/d2l/home/217890

Required work and grading:

- 7 homework assignments (60% of your grade).

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code
  - Grader is Olivia Pomerenk (op554@nyu.edu)

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code
  - Grader is Olivia Pomerenk (op554@nyu.edu)
  - Grader's office hour: Mondays at 1pm, WWH 1110

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code
  - Grader is Olivia Pomerenk (op554@nyu.edu)
  - Grader's office hour: Mondays at 1pm, WWH 1110
  - You are welcome to discuss with your colleagues, but *you've to write up your solution independently and write every line of code yourself.*

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code
  - Grader is Olivia Pomerenk (op554@nyu.edu)
  - Grader's office hour: Mondays at 1pm, WWH 1110
  - You are welcome to discuss with your colleagues, but *you've to write up your solution independently and write every line of code yourself.*
  - The first homework assignment has been posted.

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code
  - Grader is Olivia Pomerenk (op554@nyu.edu)
  - Grader's office hour: Mondays at 1pm, WWH 1110
  - You are welcome to discuss with your colleagues, but *you've to write up your solution independently and write every line of code yourself.*
  - The first homework assignment has been posted.
  - Follow the guidelines/rules provided with the homework to present and submit your solution.

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code
  - Grader is Olivia Pomerenk (op554@nyu.edu)
  - Grader's office hour: Mondays at 1pm, WWH 1110
  - You are welcome to discuss with your colleagues, but *you've to write up your solution independently and write every line of code yourself.*
  - The first homework assignment has been posted.
  - Follow the guidelines/rules provided with the homework to present and submit your solution.
  - If you find errors/typos in the slides or homework assignments, please let me know.

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code
  - Grader is Olivia Pomerenk (op554@nyu.edu)
  - Grader's office hour: Mondays at 1pm, WWH 1110
  - You are welcome to discuss with your colleagues, but *you've to write up your solution independently and write every line of code yourself.*
  - The first homework assignment has been posted.
  - Follow the guidelines/rules provided with the homework to present and submit your solution.
  - If you find errors/typos in the slides or homework assignments, please let me know.
  - Late homework policy: We understand that there are sometimes unexpected circumstances. You can hand in one homework late by 24h this semester if you send me and the grader an email at least 24h before the deadline that your work will be late.

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code
  - Grader is Olivia Pomerenk (op554@nyu.edu)
  - Grader's office hour: Mondays at 1pm, WWH 1110
  - You are welcome to discuss with your colleagues, but *you've to write up your solution independently and write every line of code yourself.*
  - The first homework assignment has been posted.
  - Follow the guidelines/rules provided with the homework to present and submit your solution.
  - If you find errors/typos in the slides or homework assignments, please let me know.
  - Late homework policy: We understand that there are sometimes unexpected circumstances. You can hand in one homework late by 24h this semester if you send me and the grader an email at least 24h before the deadline that your work will be late.
- A final (40% of grade)

Required work and grading:

- 7 homework assignments (60% of your grade).
  - These will be mixed paper&pencil and computational/programming. You hand in solutions in LaTeX and Matlab code
  - Grader is Olivia Pomerenk (op554@nyu.edu)
  - Grader's office hour: Mondays at 1pm, WWH 1110
  - You are welcome to discuss with your colleagues, but *you've to write up your solution independently and write every line of code yourself.*
  - The first homework assignment has been posted.
  - Follow the guidelines/rules provided with the homework to present and submit your solution.
  - If you find errors/typos in the slides or homework assignments, please let me know.
  - Late homework policy: We understand that there are sometimes unexpected circumstances. You can hand in one homework late by 24h this semester if you send me and the grader an email at least 24h before the deadline that your work will be late.
- A final (40% of grade)
  - Expect to write *some* code too

Organization issues

Prerequisites:

- Basic linear algebra; calculus; experience in Matlab (or Python or another programming language)

There is a part II of this class...

- ...in the Spring semester. You should take both parts to get a reasonably complete overview of Numerical Methods.
- If you consider taking only one semester of Numerical Methods, I recommend taking Scientific Computing this semester instead of this class.

Topics covered in Numerical Methods I

Numerical Methods and their Analysis

- Stability; sources of errors; error propagation, representation of numbers in computers
- Numerical linear algebra: direct solution of sparse/dense linear system; solution of least square systems; eigenvalue problems; iterative solution of linear systems
- Nonlinear systems; Newton's method; Nonlinear least squares
- Numerical optimization
- Interpolation and Approximation
- Numerical integration

Computing Issues

- What makes some computer codes faster than others?
- Where are numerical methods used, and what is their role in science research?
- How large/complicated problems can we solve today? Where are the challenges and limits of what we can do?

Topics of Numerical Methods II

Main topics covered in Numerical Methods II in the Spring semester

- Approximation of ordinary differential equations (ODEs)
- Approximation of partial differential equations (PDEs)
- Solvers for the resulting (high-dimensional) discrete problems

Programming

Programming the methods we discuss is an integral part of this course. To really understand methods & algorithms, one needs to implement them and experiment with them.

- Make sure you have access to MATLAB (CIMS, student license), you will need it for the first homework assignment.
- Alternatives to MATLAB: Octave, Python or Julia.
- We will talk about a few best coding practices, and how to present results.

Recommended textbooks/literature:

Text books:

- P. Deuflhard, A. Hohmann: *Numerical Analysis in Modern Scientific Computing. An Introduction*, 2nd edition, Springer, 2003.
- L. N. Trefethen, D. Bau: *Numerical Linear Algebra*, SIAM, 1997.
- A. Quarteroni, R. Sacco, F. Saleri: *Numerical Mathematics*, 2nd edition, Springer, 2007.
- M. Overton: Numerical Computing with IEEE Floating Point Arithmetic, SIAM, 2004.

Matlab/Programming:

- W. Gander, M. J. Gander, F. Kwok: *Scientific Computing - An Introduction Using Maple and MATLAB*. Texts in Computation Science and Engineering. Springer, 2014.
- C. Moler: *Numerical Computing with Matlab*, SIAM, 2007.

Numerical mathematics

Computer simulations have had a big influence on research and development; sometimes the ability to simulate phenomena is referred to as the third pillar of science.

Numerical mathematics is a part of mathematics that develops, analyzes and applies methods from scientific computing to

- analysis
- linear algebra
- optimization
- differential equations
- ...

It has applications accross many applied sciences, including:

- physics
- economics
- biology
- finance
- ...

Development of Numerical Methods at Courant

A few examples...

- Eigenvalue problems (Overton)
- Fast multipole method (Greengard, O'Neil, Zorin)
- Immersed boundary method for solid-fluid interactions (Peskin)
- Methods for studying dynamical systems, multiscale methods (Vanden-Eijnden)
- Methods for free boundary problems in fluid dynamics (Shelley)
- Scalable implicit solvers for viscous flows (Donev, Stadler)
- Sampling methods and Uncertainty Quantification (Goodman, Stadler, Peherstorfer)
- Scientific machine learning (Vanden-Eijnden, Stadler, Peherstorfer)
- ...

Applications of Numerical Methods at Courant

A few examples...

- Simulation and analysis of natural and artificial heart valves (Peskin)
- Simulation of plate tectonics and mantle convection (Stadler)
- The physics of cell's interiors and their motion (Shelley)
- Computational fluid/hydrodynamics (Donev)
- Optimal complexity wave simulations (Greengard)
- Simulation of blood cells-resolving blood flow (Zorin)
- Plasmas (Stadler)
- ...

Seminars

Numerical Analysis and Scientific Computing seminar

- Fridays at 10:00, WWH 1302
- Talks about current research
- https://cs.nyu.edu/dynamic/news/seminars/1/

SEP 16

10AM, Warren Weaver Hall 1302

Bilevel learning for inverse problems

Juan Carlos De los Reyes, Escuela Politécnica Nacional

Update Delete

SEP 30

10AM, Warren Weaver Hall 1302

Randomized matrix-free quadrature

Tyler Chen, CIMS

Update Delete

OCT 7

10AM, Online

Derivative-Free Bayesian Inference for Large Scale Inverse Problem

Daniel Zhengyu Huang, California Institute of Technology

Update Delete

Modeling and Simulation meeting

- Thursdays at 12:30, WWH 1302
- Student-driven meeting on topics related to computational mathematics
- https://math.nyu.edu/dynamic/research/pages/research-and-training-group-mathematical-modeling-and-simulation/activities/group-meeting/

Modeling and Simulation meeting

- Thursdays at 12:30, WWH 1302
- Student-driven meeting on topics related to computational mathematics
- https://math.nyu.edu/dynamic/research/pages/research-and-training-group-mathematical-modeling-and-simulation/activities/group-meeting/

Mathematics colloquium

- Mondays at 3:45, WWH 1302
- https://math.nyu.edu/dynamic/calendars/seminars/mathematics-colloquium/

Modeling and Simulation meeting

- Thursdays at 12:30, WWH 1302
- Student-driven meeting on topics related to computational mathematics
- https://math.nyu.edu/dynamic/research/pages/research-and-training-group-mathematical-modeling-and-simulation/activities/group-meeting/

Mathematics colloquium

- Mondays at 3:45, WWH 1302
- https://math.nyu.edu/dynamic/calendars/seminars/mathematics-colloquium/

Math and data

- Thursdays at 2.00, Auditorium Hall 150, Center for Data Science, NYU, 60 5th ave.
- Interface of Applied Mathematics, Statistics and Machine Learning
- https://mad.cds.nyu.edu/seminar/

Outline

Organization

Conditioning of problems

Stability of algorithms

Representing real numbers

Condition of a problem

- Consider a generic problem: given F and data/input x, find output y such that

$$F(x, y) = 0$$

- Let's assume there is a unique solution so that we can write

$$y = f(x)\,,$$

  for a function f in the following
- Well-posed: Unique solution + If we perturb the input x a little bit, the solution y gets perturbed by a small amount.
- Otherwise, the problem is ill-posed; no numerical method can help with that. (What should we do in such a situation?

Condition of a problem

- Consider a generic problem: given F and data/input x, find output y such that

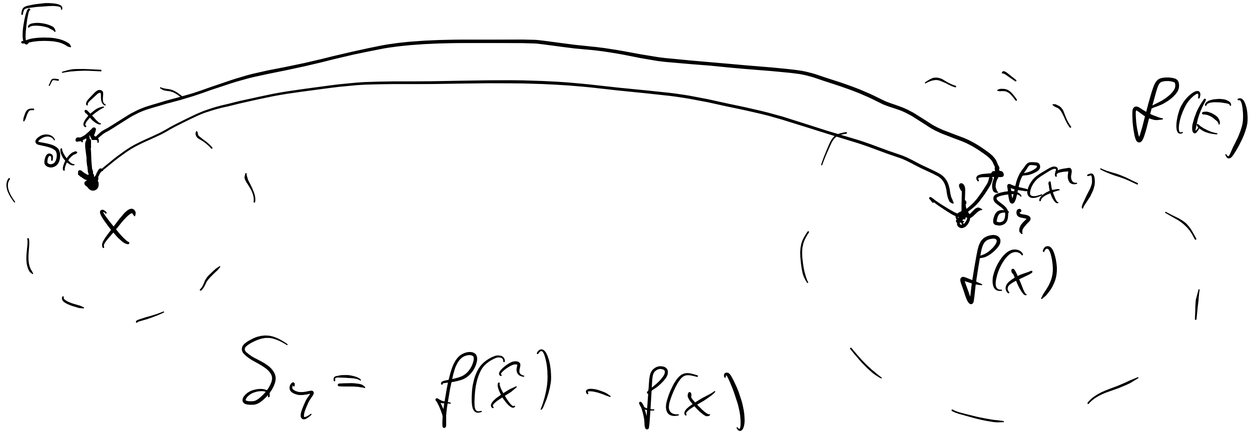
$$F(x, y) = 0$$

- Let's assume there is a unique solution so that we can write

$$y = f(x)\,,$$

  for a function f in the following
- Well-posed: Unique solution + If we perturb the input x a little bit, the solution y gets perturbed by a small amount.
- Otherwise, the problem is ill-posed; no numerical method can help with that. (What should we do in such a situation? $\rightsquigarrow$ change the problem)

Condition of a problem (visualization)



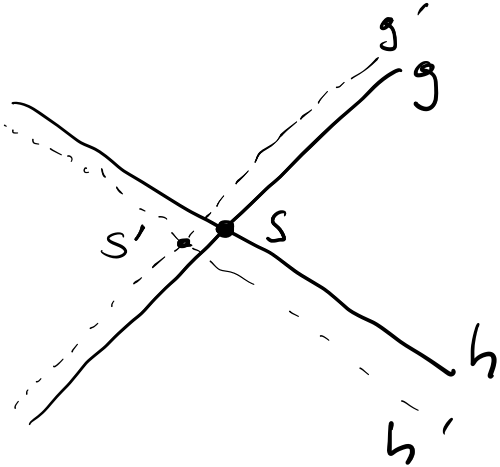
$$\delta_y = f(\hat{x}) - f(x)$$

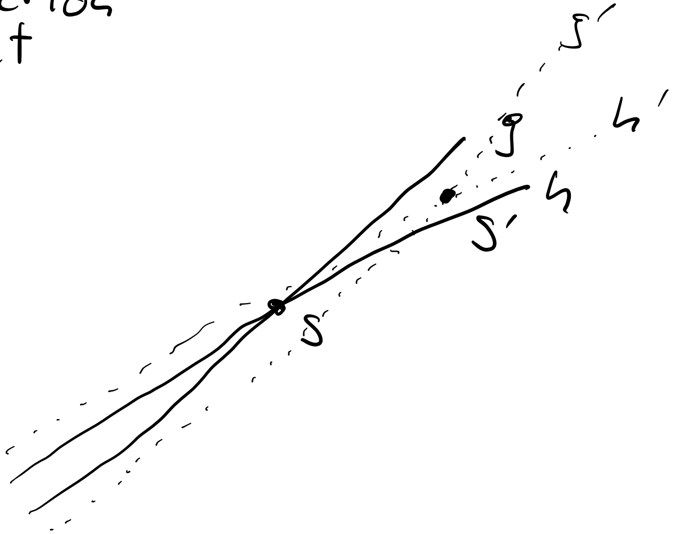
Condition of a problem (intersecting lines)

$f: (g, h) \mapsto S$

two lines

intersection point





Condition of a problem (cont'd)

- Terms such as "little bit" and a "small amount" already point to that we need to measure something
- Therefore, we assume the map f is given as

$$f : U \subset \mathbb{R}^n \to \mathbb{R}^m$$

  and we are interested in the norm $\|\cdot\|$
- The input error is then

$$\|x - \hat{x}\| \leq \delta \text{ (absolute)} \qquad \|x - \hat{x}\| \leq \delta \|x\| \text{ (relative)}$$

- Correspondingly we measure the output error $f(x) - f(\hat{x})$ in $\|\cdot\|$ (we could also have looked at a componentwise error)

Condition of a problem (cont'd)

- Absolute condition number at x is

$$\kappa_{\text{abs}} = \lim_{\delta \to 0} \sup_{\|x-\hat{x}\| \le \delta} \frac{\|f(x) - f(\hat{x})\|}{\|x - \hat{x}\|}$$

- Relative condition number at x is

$$\kappa_{\text{rel}} = \lim_{\delta \to 0} \sup_{\|x-\hat{x}\| \le \delta} \frac{\|f(x) - f(\hat{x})\| / \|f(x)\|}{\|x - \hat{x}\| / \|x\|}$$

- If f is differentiable in x, then

$$\kappa_{\text{abs}} = \|f'(x)\| \qquad \kappa_{\text{rel}} = \frac{\|x\|}{\|f(x)\|} \|f'(x)\|,$$

where $\|f'(x)\|$ is the norm of the Jacobian $f'(x)$ in the operator norm

$$\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \sup_{\|x\|=1} \|Ax\|$$

- Another way to interpret the condition number at x is via the bounds

$$\|f(\hat{x}) - f(x)\| \leq \kappa_{\mathsf{abs}} \|\hat{x} - x\|$$

and

$$\frac{\|f(\hat{x}) - f(x)\|}{\|f(x)\|} \leq \kappa_{\mathsf{rel}} \frac{\|\hat{x} - x\|}{\|x\|},$$

for infinitesimal δ (or $\hat{x} \to x$)

Condition of a problem (cont'd)

- If $\kappa_{\text{rel}} \sim 1$,

Condition of a problem (cont'd)

- If $\kappa_{\mathrm{rel}} \sim 1$, then the problem is well conditioned: If the relative error in the data/input is small, then the relative error in the answer/output is similarly small
- If $\kappa_{\mathrm{rel}} \gg 1$,

Condition of a problem (cont'd)

- If $\kappa_{\text{rel}} \sim 1$, then the problem is well conditioned: If the relative error in the data/input is small, then the relative error in the answer/output is similarly small
- If $\kappa_{\text{rel}} \gg 1$, then the problem is poorly conditioned: Small relative input error can lead to large relative output error
- If κ_{rel} (and κ_{abs}) do not exist, then the problem is ill conditioned.

- What is poorly conditioned depends on desired accuracy: if the input accuracy is low but we expect a high output accuracy, then problems are quickly poorly conditioned. If we are happy with a less accurate output, we might consider the problem still well conditioned.
- Sometimes, the possibly large error in the output does not matter and so we can solve poorly conditioned problems (think of early design stages, rapid prototyping, etc); but we should be very much aware of the condition of the problem.

Condition of a problem: Example

Condition of „addition"

$$f : \mathbb{R}^2 \to \mathbb{R}, \quad (a,b) \mapsto f(a,b) = a+b$$

derivative

$$f'(a,b) = [1, 1] \in \mathbb{R}^{1\times 2}$$

choose 1-norm

$$\left\| \begin{bmatrix} a \\ b \end{bmatrix} \right\|_1 = |a| + |b|$$

$$\| f'(a,b)\|_1 = \| [1, 1] \|_1 = 1$$

$$k_{abs} = \| f'(a,b)\|_1 = 1$$

$$k_{rel} = \frac{\|x\|}{\|f(x)\|} \| f'(x)\| = \frac{|a| + |b|}{|a+b|} \cdot 1$$

if $a \cdot b > 0 \Rightarrow k_{rel} = 1$

$a \cdot b < 0$

$a > 0$, $b = -(a+\varepsilon)$, $0 < \varepsilon \ll 1$

$$k_{rel} = \frac{a + a + \varepsilon}{|a - (a+\varepsilon)|} = \frac{2a+\varepsilon}{\varepsilon} \gg 1$$

$a = 0.1234\underline{6}7$

$b = 0.1234\underline{5}6$

$a - b = 0.0000\underline{1}1$

Mat Vec: $x \mapsto Ax$

$$f(x) = Ax$$

$$f'(x) = A$$

$$k_{abs} = \| f'(x) \| = \| A \|$$

$$k_{rel} = \frac{\|x\|}{\|Ax\|} \|A\|$$

take $\| \cdot \|_2$

$$\|A\|_2 = \sigma_{max}(A)$$

Condition of linear systems

$$Ax = b$$

(i) Problem: fix A, $b \mapsto A^{-1} b$

$$f: \mathbb{R}^n \to \mathbb{R}^n, \quad f(b) = A^{-1} b$$

$$k_{abs} = \| f'(b) \| = \| A^{-1} \|$$

$$k_{rel} = \frac{\|b\|}{\|A^{-1} b\|} \|A^{-1}\|$$

$$A^{-1} b = x$$

$$\kappa_{rel} = \frac{\|Ax\|}{\|x\|}\|A^{-1}\| \leq \frac{\|A\|\ \|x\|}{\|x\|}\|A^{-1}\|$$

$$= \|A\|\ \|A^{-1}\|$$

$$= \kappa(A)$$

if we take $\|\cdot\|_2$ on input b, then operator norm

$$\|A^{-1}\|_2 = \sigma_{max}(A^{-1}) = \frac{1}{\sigma_{min}(A)}$$

$$\kappa_2(A) = \frac{\sigma_{max}(A)}{\sigma_{min}(A)}$$

Problem: fix b, consider $A \mapsto A^{-1}b$

$$f(A) = A^{-1}b$$

$$\|f'(A)\| \leq \|A^{-1}\|\ \|x\|$$

$$x = A^{-1}b$$

$$\kappa_{abs} = \|f'(A)\| \leq \|A^{-1}\|\ \|x\|$$

$$\kappa_{rel} = \frac{\|A\|}{\|f(A)\|}\|f'(A)\|$$

$$\leq \frac{\|A\|}{\|x\|}\|A^{-1}\|\ \|x\| = \|A\|\ \|A^{-1}\| = \kappa(A)$$

Outline

Organization

Conditioning of problems

Stability of algorithms

Representing real numbers

Stability of algorithms

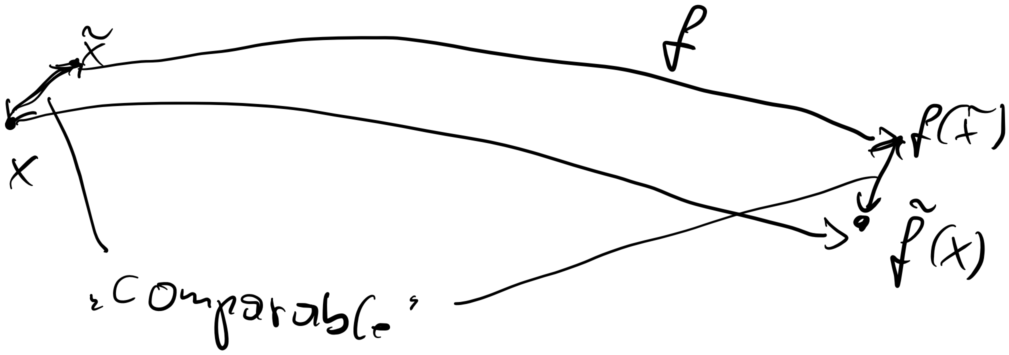
Is $\tilde{f}(x)$, computed with an algorithm $\tilde{f}$, a good approximation of $f(x)$?

We are happy if the error due to the algorithm

$$\tilde{f}(x) - f(x)$$

lies within reasonable bounds of the error due to the input

$$f(\tilde{x}) - f(x)$$


f
$\tilde{x}$
x
$f(\tilde{x})$
$\tilde{f}(x)$
„comparable"

Stability: Stability

We say that an algorithm $\tilde{f}$ for a problem f is stable if for each $x \in E$ the error

$$\frac{\|\tilde{f}(x) - f(\tilde{x})\|}{\|f(\tilde{x})\|}$$

is small for some $\tilde{x}$ with

$$\frac{\|\tilde{x} - x\|}{\|x\|}$$

small

> *A stable algorithm gives nearly the right answer ($\tilde{f}(x)$) to nearly the right question ($f(\tilde{x})$).*

In forward error analysis one tries to establish stability by showing error bounds on the result in each operation in the algorithm in order to bound the error in the end result

Stability: Backward stability

Backward stability: Pass the errors of the algorithm back and interpret as input errors.

An algorithm $\tilde{f}$ for a problem f is backward stable if for each $x \in X$ we have $\tilde{f}(x) = f(\tilde{x})$ for an $\tilde{x}$ with

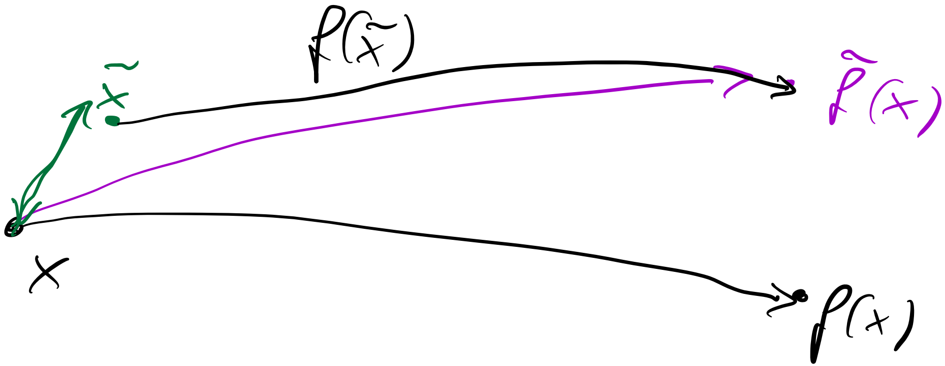
$$\frac{\|\tilde{x} - x\|}{\|x\|}$$

small

This is a tightening of the definition of stability of the previous slide:

> *A backward stable algorithm gives exactly the right answer to nearly the right question.*

In backward error analysis one calculates, for a given output, how much one would need to perturb the input in order for the answer to be exact.


$\tilde{x}$
$f(\tilde{x})$
$\tilde{f}(x)$
x
$f(x)$

Errors and error analyses

Relative errors:

$$\frac{\|x - x_n\|}{\|x\|}$$

Absolute error:

$$\|x - x_n\|$$

- Used for theoretical arguments
- In numerical practice: exact solution is not available, so these errors must be approximated.

A priori analysis is performed before a specific solution is computed. Typically, the analysis is performed for a large class of possible inputs.
A posteriori analysis bounds the error for a specific numerical solution $\hat{x}$ (computed with a specific numerical method), and uses, e.g., residuals for the a posteriori analysis.

Computational errors

Numerical algorithms try to control or minimize, rather then eliminate, the various computational errors:

- **Approximation error** due to replacing the computational problem with an easier-to-solve approximation. Also called discretization error for ODEs/PDEs.
- **Truncation error** due to replacing limits and infinite sequences and sums by a finite number of steps. Closely related to approximation error.
- **Roundoff error** due to finite representation of real numbers and arithmetic on the computer, $x \neq \hat{x}$.
- **Propagated error** due to errors in the data from user input or previous calculations in iterative methods.
- **Statistical error** in stochastic calculations such as Monte Carlo calculations.

Intuition: Stability, Consistency, Convergence

Instead of solving $F(x, y) = 0$ directly, many numerical methods generate a solution sequentially

$$\bar{F}(x_i, x_{i-1}) = 0\,, \qquad i = 1, 2, 3, \dots\,,$$

with $x_0 = x$ and sequence (x_i) converging to y

Additionally, we use a numerical method $\hat{F}_n$ instead of $\bar{F}$

$$\hat{F}_n(\hat{x}_i, \hat{x}_{i-1}) = 0\,, \qquad i = 1, 2, 3, \dots\,,$$

with method $\hat{F}_n$ depending on a parameter n: Increasing n typically means investing more computational time for a hopefully more accurate result

Consistent: A numerical method is consistent if the local error made at each step vanishes for $n \to \infty$

$$\hat{F}_n(x_i, x_{i-1}) \to \bar{F}(x_i, x_{i-1}) \qquad (n \to \infty)$$

This is one of the most basic requirements that we have on a numerical approach. If it is not consistent, it means we can invest more computational time (more effort) and certainly won't get lower errors.

Stability: Because we use $\hat{F}_n$ instead of $\bar{F}$, in each iteration we make a local error (see above). We have $\hat{x}_i$ at iteration i rather than x_i. Stability means here that the local error can be amplified only by a constant that is independent of n.

Convergence: If the numerical error can be made arbitrarily small by increasing the computational effort $n \to \infty$

$$\boxed{\text{consistency} + \text{stability} \to \text{convergence}}$$

A concrete and formal description of these concepts for finite difference approximations can be found in Chapter 2 of LeVeque's textbook on finite difference methods.

Speed of convergence

Let $x_n \to x$ in a normed space $X, \|\cdot\|$ for $n \to \infty$.

$$\lim_{n\to\infty} \frac{\|x - x_{n+1}\|}{\|x - x_n\|^q} \leq C$$

with $C > 0$ and $q \geq 1$

- Linear convergence: $q = 1$ and $C < 1$

$$\|x - x_{n+1}\| \leq C\|x - x_n\|$$

- Quadratic convergence: $q = 2$

$$\|x - x_{n+1}\| \leq C\|x - x_n\|^2$$

Beyond convergence

- An algorithm will produce the correct answer if it is convergent, but...
- Not all convergent methods are equal. We can differentiate them further based on:

Beyond convergence

- An algorithm will produce the correct answer if it is convergent, but...
- Not all convergent methods are equal. We can differentiate them further based on:
- **Accuracy** How much computational work do you need to expand to get an answer to a desired relative error?

Beyond convergence

- An algorithm will produce the correct answer if it is convergent, but...
- Not all convergent methods are equal. We can differentiate them further based on:
- **Accuracy** How much computational work do you need to expand to get an answer to a desired relative error?
- **Robustness** Does the algorithm work (equally) well for all (reasonable) input data d?

Beyond convergence

- An algorithm will produce the correct answer if it is convergent, but...
- Not all convergent methods are equal. We can differentiate them further based on:
- **Accuracy** How much computational work do you need to expand to get an answer to a desired relative error?
- **Robustness** Does the algorithm work (equally) well for all (reasonable) input data d?
- **Efficiency** How fast does the implementation produce the answer? This depends on the algorithm, on the computer, the programming language, the programmer, etc. (more next class)

Beyond convergence

- An algorithm will produce the correct answer if it is convergent, but...
- Not all convergent methods are equal. We can differentiate them further based on:
- **Accuracy** How much computational work do you need to expand to get an answer to a desired relative error?
- **Robustness** Does the algorithm work (equally) well for all (reasonable) input data d?
- **Efficiency** How fast does the implementation produce the answer? This depends on the algorithm, on the computer, the programming language, the programmer, etc. (more next class)
- **Difficulty** How easy is it to implement and apply in practice? Do I need to spend 5 years of my time to implement it or can I code it up in 2 lines of code?