Heidelberg University

Department of Physics and Astronomy

Book in Progress

An Introduction to Computational Physics

Leonard Storcks

February 7, 2024

Abstract

Computation - quickly crunching billions of numbers - can give us a new perspective and understanding of physical systems. This book aims to equip the reader with fundamental knowledge of numerics, computation and statistics as well as tools to tackle problems from physics (and many other disciplines).

Contents

1 Introduction

Add / refer to example applications

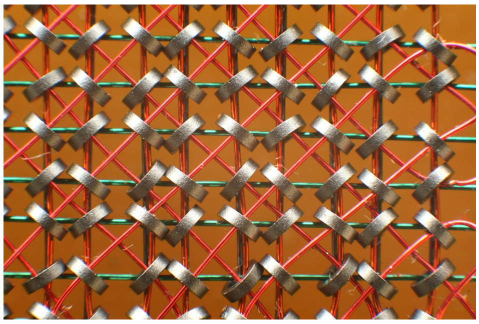
Computational physics encompasses

- simulation as a new paradigm to approach physical problems and validate theories by comparing simulated results to experiments
- statistics and data analysis to make sense of experimental or simulated data
- scientific machine learning to incorporate physical knowledge into Machine Learning and Machine Learning into simulations

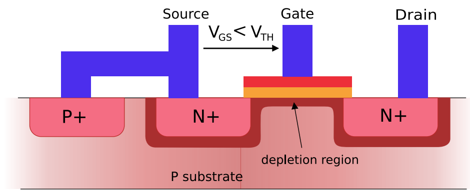
This books aim is to give an introduction to Computational Physics with the presented concepts often also being of great use in other fields (e.g. solving partial differential equations computationally in not only greatly important for physics but for instance also for solving the Black-Scholes equation in finance).

Content included mainly encompasses the lectures

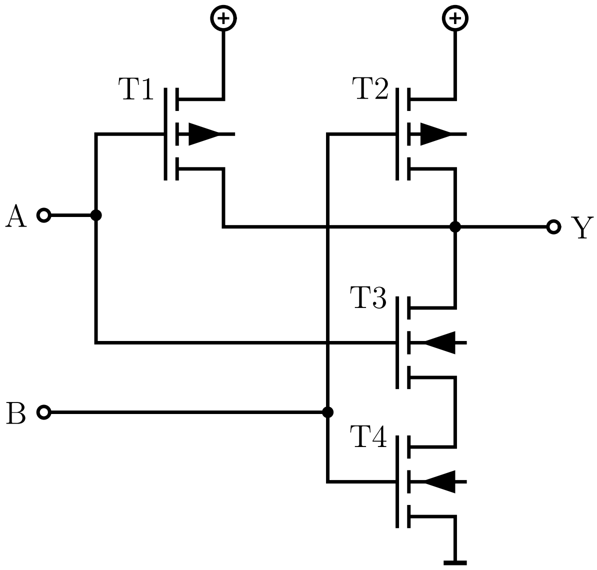
- Fundamentals of Simulation Methods (Ralf Klessen and Philip Girichidis, 2023)
- Computational Statistics and Data Analysis (Daniel Durstewitz, 2023)



(a) Historic Magnetic Core Storage. Bits are stored as the direction of magnetic flux.



(b) Field-effect transistor (more specifically MOSFET schematic). Power-efficiently switching currents is at the heart of modern computing. Based on a Floating-Gate or Charge-Trapping mechanism and the tunnel effect, storage can be realized via stored charges.



(c) NAND-Gate. Bit operations lie at the core of computations.

Figure 1: Basics of Computation.

Part I

Basics of Numerical Computation

How we write what algorithms is informed by how we represent data. While there exist exotic approaches like analog computers that can e.g. handle integration elegantly (Ulmann, 2020) or quantum computing which might e.g. at some point accelerate machine learning (Biamonte et al., 2017), standard binary data representation is vastly prevailing.

Binary data can be very efficiently and reliably stored and operated on (see figure 1), but the respective chosen representations might come with caveats.

2 Digital Representation of Numbers

In C, `int x = 100 * 200 * 300 * 400;` will surprisingly yield -1894967296 (for a 32-bit integer representation) (*overflow*), `float f1 = (2.3 + 1e20) - 1e20;` yields 0.0 (*round-off-error*) but `float f2 = 2.3 + (1e20 - 1e20);` correctly gives 2.3.

We want to understand and mitigate such caveats of arithmetic on computers, where we want quick calculations while also using as little storage as possible - simulations can quickly become big in storage (e.g. lots of particles).

Further details can be found in Higham, 2002 and Bryant and O'Hallaron, 2011.

2.1 Integer Arithmetic

In C, integers ($\in \mathbb{Z}$) are stored as fixed-size bit-sequences. The respective size dictates a range. C provides both unsigned and signed integer types, with negative numbers in the signed case represented using *two's-complements*.

2.1.1 Unsigned integers

For a bit-vector $\underline{x} = [x_{\omega-1}, x_{\omega-2}, \dots, x_0]$ the unsigned conversion is

$$B2U_\omega := \sum_{i=0}^{\omega-1} x_i 2^i \tag{1}$$

(illustrated in figure 2) and the range is $0, \dots, 2^\omega - 1$. In case of overflow in operations, the overflow is truncated in the most significant bits (see figure 3). As $B2U_\omega[x_{\omega-1}, x_{\omega-2}, \dots, x_0] \mod 2^k = B2U_\omega[x_{k-1}, x_{k-2}, \dots, x_0]$ we effectively store arithmetic result $\mod 2^\omega$.

$x_{\omega-1}$ is called most significant bit (MSB) and x_0 least significant bit (LSB).

Problem: When the result of an arithmetic operation exceeds the range of an integer type, unexpected results occur (overflow) (and also underflow in the signed case)

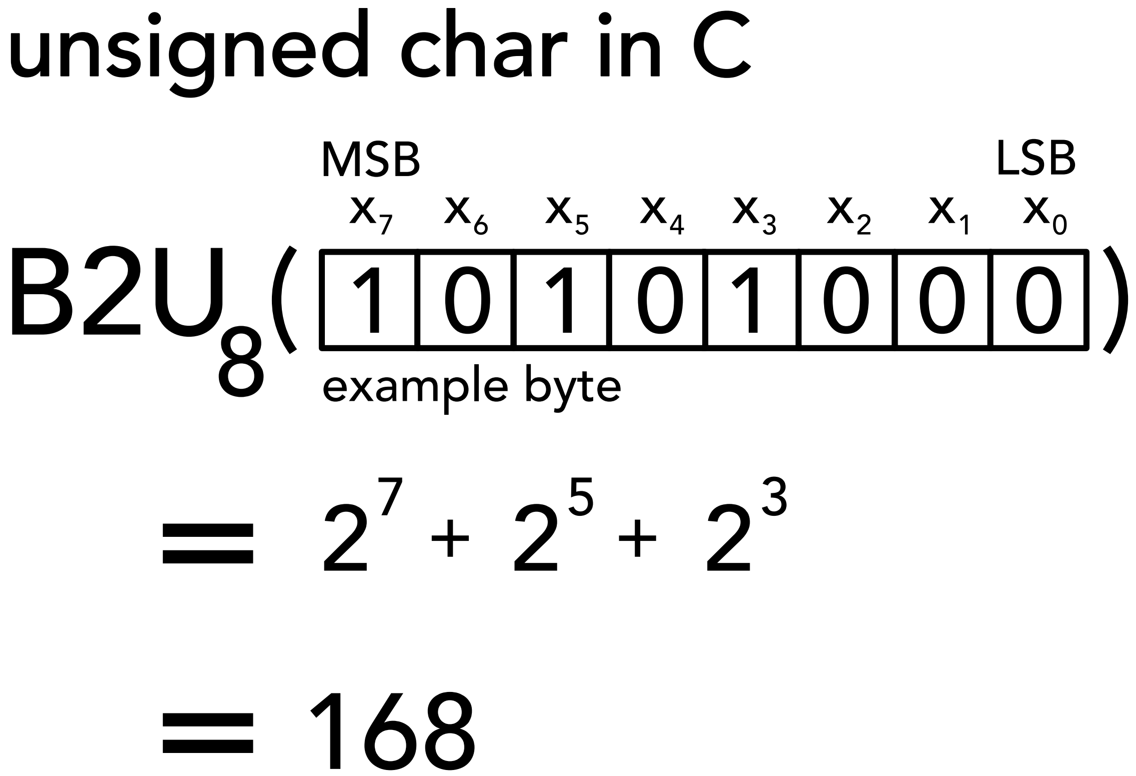


Figure 2: Example of an unsigned char in C.

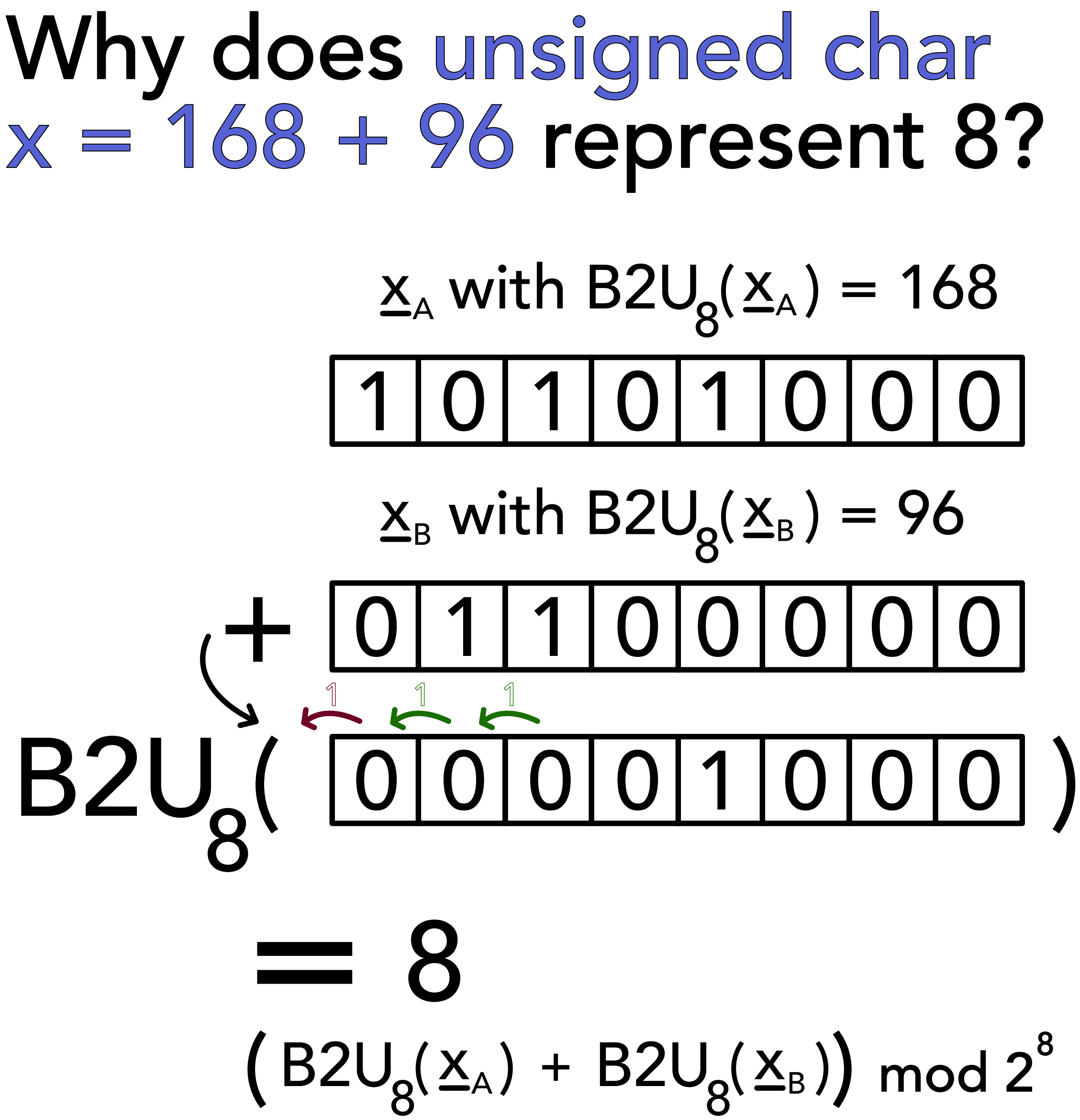


Figure 3: Example of unsigned char overflow.

2.1.2 Two's complement for negative numbers

Note that addition of integers by bitwise addition with carry-on is very fast.

Why can't we just let the MSB encode the sign?: A representation of the form

$$B2S_\omega(\underline{x}) := (-1)^{x_{\omega-1}} \cdot \sum_{i=0}^{\omega-2} x_i 2^i \tag{2}$$

(sign magnitude) has the disadvantage, that normal bitwise addition with carry-on does not work. Also, zero is encoded twice, as ± 0.

Idea of the two's-complement: The MSB flags the sign in $\underline{x} = [x_{\omega-1}, x_{\omega-2}, \ldots, x_0]$ by having a weighting factor $-2^{\omega-1}$

$$B2T_\omega(\underline{x}) := -x_{\omega-1}2^{\omega-1} + \sum_{i=0}^{\omega-2} x_i 2^i \tag{3}$$

Carry-on to the ω-th bit in addition is again ignored. As we really just have to add positive numbers in bits x_0 to $x_{\omega-2}$ with correct sign-switch by carry-on, no special handling is necessary (see figure 4). The range is $-2^{\omega-1} \ldots 2^{\omega-1} - 1$.

From an unsigned int to the two's complement and vice versa we can get by

1. invert all bits
2. add +1 to the result[1]

If we want $\underline{x}$ with $B2T_\omega(\underline{x}) = -u$ then the bits following the MSB must encode k with $-u = -2^{\omega-1} + k$, so the encoded unsigned number must be $B2T_\omega(\underline{x}) = \underbrace{2^{\omega-1}}_{\text{sign bit}} + k = 2^\omega - u$ - a *two's complement*.

[1] These rules intuitively follow from the constraint, that bitwise addition with carry-on of $-u$ and u should result in an all-zero bitvector. Adding a bitvector to its inverted self results in an all-1 bitvector, adding one more then results in all zeros, as the last carry-on is discarded.

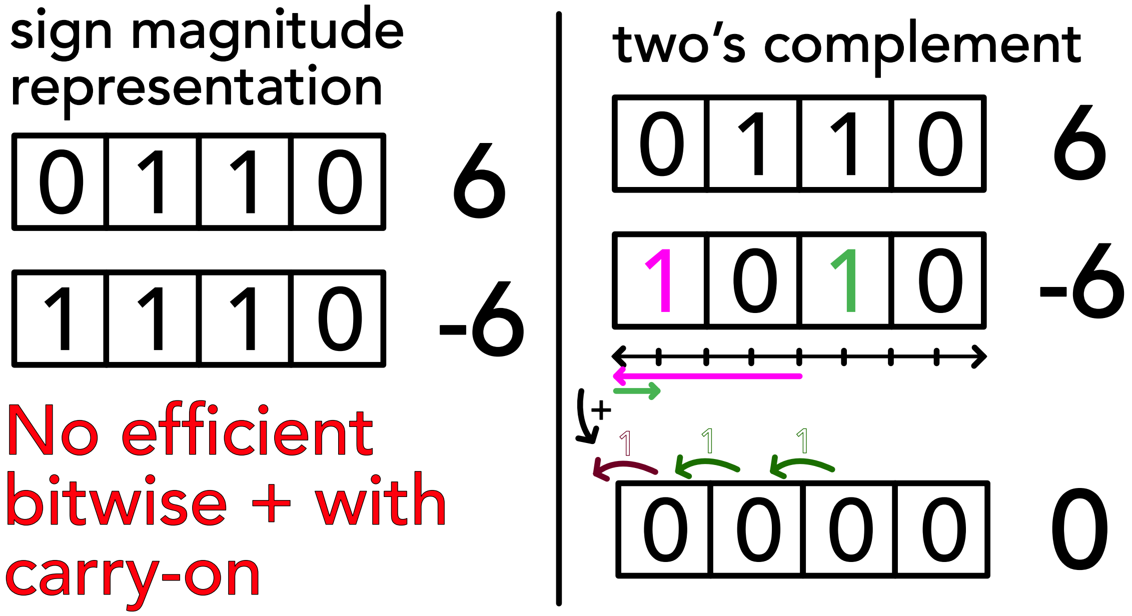


Figure 4: Illustration of the defining feature of the two's complement - addition is simple.

2.1.3 Integer types in C

Some common integer types with respective minimum sizes are given in table 1.

Type	Minimum Size ω
`char`	8 bits
`short`	16 bits
`int`	16 bits
`long`	32 bits
`long long`	64 bits

Table 1: Common integer types in C, signed ranges are $-2^{\omega-1} \dots 2^{\omega-1}-1$, unsigned ranges are $0 \dots 2^{\omega}-1$.

Problem: Using unsigned can come with unexpected results, when being cast from a negative number. E.g. `unsigned int x = -1;` will result in $2^{32}-1$ in a 32-bit system (the two's complement is read as if an unsigned representation). More deviously, $-1 < 0U$ will evaluate to false, as all integers in a comparison are cast to unsigned, when one of them is unsigned.

2.1.4 Byte Ordering in Storage: Big and Little Endian

Bytes of e.g. a multi-byte integer can be stored from most significant byte to least significant byte (big endian) or vice versa (little endian), see figure 5.

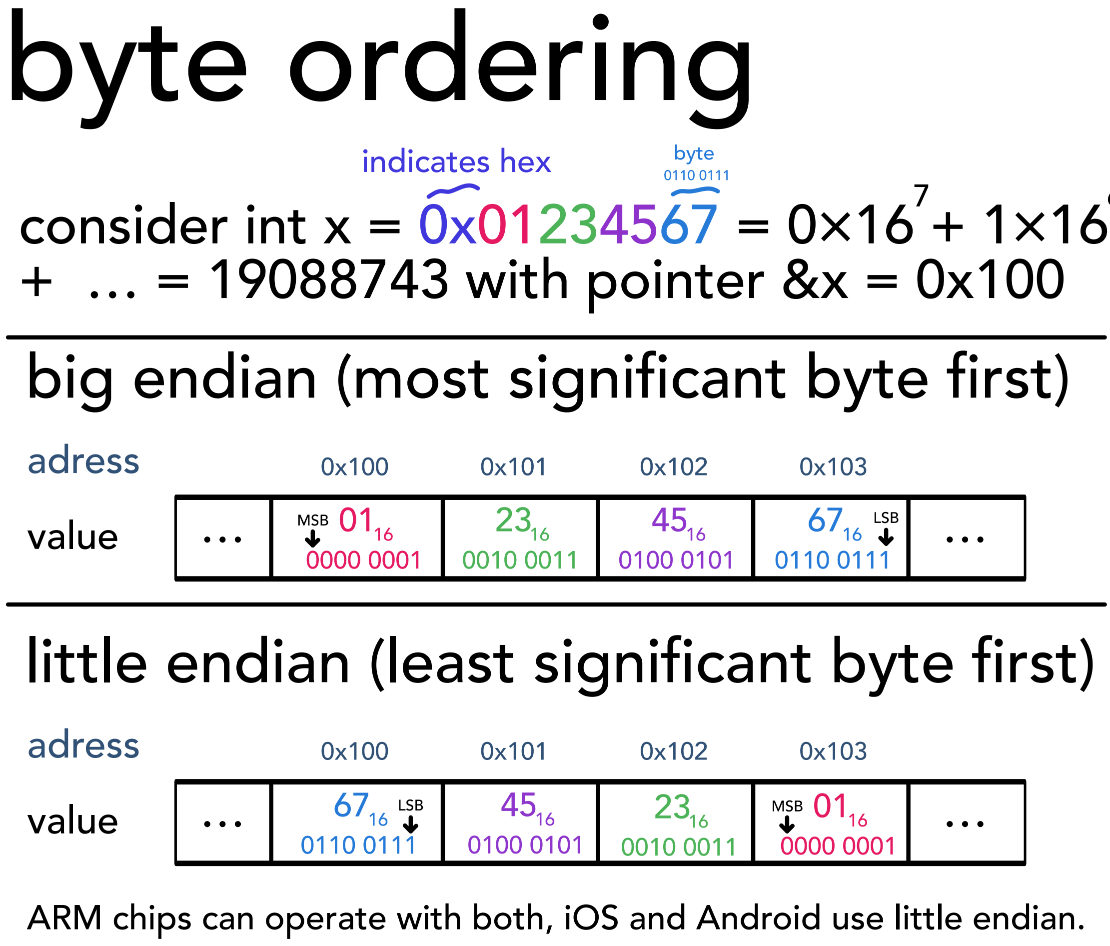


Figure 5: Big and Little Endian.

2.1.5 Properties and Caveats of Integer Arithmetic

Typical problems in integer arithmetic are overflow, integer division, the modulo operation and implicit type conversion.

Overflow: The respective integer ranges have finite ranges, overflow in arithmetic operations is cut-off. We must choose a type with sufficient range - mind that choosing types with too large of a footprint (e.g. always long) wastes storage and compute.

Example I: In `char c = 100 * 4; // range -128 to 127` the result is -112 as 400 in bits is 000110010000 where a cut-off to one byte means 10010000 so $-2^7+2^4 = -128+16 = -112$.

Example II: For `int a = -1 * pow(2,31); int b = 10;` we have $a < b$ but not $a - b < 0$ (for char the behavior is a bit different as up to int there is implicit type conversion.)

Integer division: All decimal places are truncated, so

$$5/3 = 1; -5/3 = -1; 5/-3 = -1$$

Modulo operation: The modulus in C is defined as $n\%m = n - (n/m)m$ so

$$5\%3 = 2; -5\%3 = -2; 5\% - 3 = -2$$

Implicit type conversion: In C, the type can implicitly be converted up to unsigned int, which can avoid overflow, as illustrated in table 2.

implicit type conversion up to int can be helpful	mind its only up to int	explicit type consversion

<pre>char a,b;
a = 100;
b = 4;
int c = a * b; // 400</pre>
	<pre>int a = 2e9;
	int b = 3;
	long long c = a * b;
	printf("%llu\n", c); // 1705032704</pre> As $2^{32} - 1 \approx 4 \cdot 10^9 < 6 \cdot 10^9 < 2^{33} - 1$ (unsigned ranges) we overflow to the 33rd-bit, which is cut-off, giving the unsigned result of $6 \cdot 10^9 - 2^{32} = 1705032704$.
		<pre>int a = 2e9;
		int b = 3;
		long long c = ((long long) a) * ((long long) b);
		printf("%llu\n", c); // 6000000000</pre>

Table 2: Type conversion and its caveats

2.1.6 Can there be integer-overflow in python?

Note that in python3, integers are implemented as “long” integer objects of arbitrary size, overflows are impossible (at the cost of speed; mind that e.g. numpy is based on C code).

2.2 Floating Point Arithmetic

In the following, we will encode rational numbers in the form $V = x \cdot 2^y$, with $x, y \in \mathbb{Z}$. Similar to the decimal notation

$$d_m d_{m-1} \dots d_0 . d_{-1} d_{-2} \dots d_{-n} = \sum_{i=-n}^{m} d_i 10^i \tag{4}$$

we can write in binary notation

$$b_m b_{m-1} \dots b_0 . b_{-1} b_{-2} \dots b_{-n} = \sum_{i=-n}^{m} b_i 2^i, \quad 0.01_2 = 0.25_{10} \tag{5}$$

or generally in base β in scientific notation

$$\begin{gathered}(-1)^s \cdot \underbrace{b_0 . b_{-1} b_{-2} \dots b_{-n}}_{\text{mantissa M}} \cdot \beta^e = \beta^e \cdot \sum_{i=-n}^{0} b_i 2^i, \qquad \text{exponent } e, \\ \text{precision (number of digits) } p = n+1, \qquad \text{sign-bit } s \in \{0,1\}\end{gathered} \tag{6}$$

Note that in this notation (in the form $V = x \cdot 2^y$) we cannot exactly represent e.g. 0.1 or 0.2.

$$0.1_{10} = 1.10011[0011]\dots_2 \cdot 2^{-4} \tag{7}$$

Disastrous historic example: In the first Gulf War (more specifically on 25th February 1991), a Patriot missile defense battery failed to intercept an incoming Iraqi Scud missile, because it used an internal clock counting up in tenths of a second represented by a 23-bit sequence. Future missile positions are predicted by extrapolating from past position with constant velocity. The Patriot system mixed both this inaccurate internal clock and a more accurate one, leading to the failed interception and 28 deaths among soldiers.

2.2.1 IEEE 754 Floating Point Standard

Based on the representation in scientific notation, we can store a floating point number in a bit-vector with

- a sign bit s
- an exponent e stored as an unsigned integer $E = e + b$ with bias b
- a mantissa M

where for single precision (32-bit)

- the exponent is stored in 8 bits, $b = 127$, $e_{min} = -126$, $e_{max} = 127$ with $E = 255$ and $E = 0$ reserved for special cases
- the mantissa is stored in 23 bits, with the first bit being implicitly 1 (**normalization**), which can always be assumed by appropriately choosing the exponent (floating point representations are not unique), so we have $p = 23(+1)$ bits of precision encoding an integer M but need a special representation for 0

where based on the exponent we differentiate between

- **normalized values for** $1 \leq E \leq 254$ with value of the floating point number

$$
\begin{aligned}
V = (-1)^s \cdot \left(1 + \frac{M}{2^p}\right) \cdot 2^{E-b}, \quad \text{sign bit } s \\
\text{biased exponent } E = e + b, \quad \text{integer representation of mantissa } M \\
\text{precision } p = \#\text{mantissa bits} + 1 \text{ implicit bit}
\end{aligned} \tag{8}
$$

- **denormalized values for** $E = 0$ with value of the floating point number

$$
V = (-1)^s \cdot \frac{M}{2^p} \cdot 2^{-b+1}, \quad \text{for } M = 0, V = \pm 0 \text{ depending on } s \tag{9}
$$

  The denormalized numbers start just below the normalized ones (by the factor 2^{-b+1} with $+1$ as we do not have the implicit leading 1 here) and now no normalization (implicit starting 1-bit) is assumed. This allows to approach zero with gradually decreasing precision (and even spacing) (smaller numbers occupy fewer digits as of the leading zeros) and ensures that for $x \neq y$, $x - y$ is non-zero, so $\frac{1}{x-y}$ is safe for $x \neq y$.

- $\pm\infty$ **for** $E = 255$ **and** $M = 0$
- **NaN (Not a Number) for** $E = 255$ **and** $M \neq 0$

Why is the exponent stored in a biased way, not two's complement?: In a two's complement representation of the exponent to compare two numbers, we have to compare the exponents and mantissas separately and the exponent-comparison is a bit more complicated than comparing biased representations. In the biased exponent representation we can just compare the bitvectors of exponent followed by mantissa interpreted as integers (also mind the sign).

The cases are illustrated in figure 6, with a specific example in figure 7.

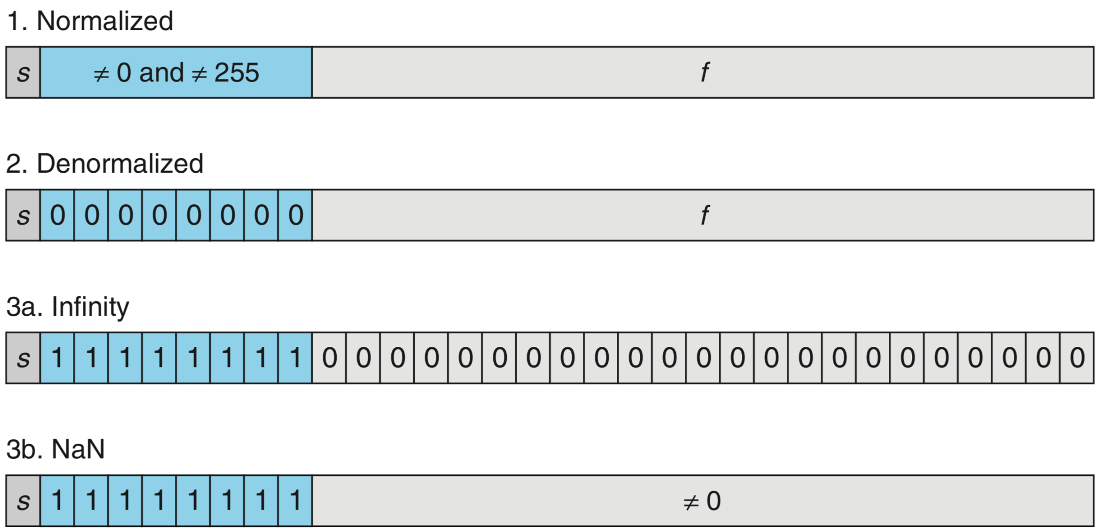


Figure 6: Cases of the IEEE 754 Floating Point Standard.

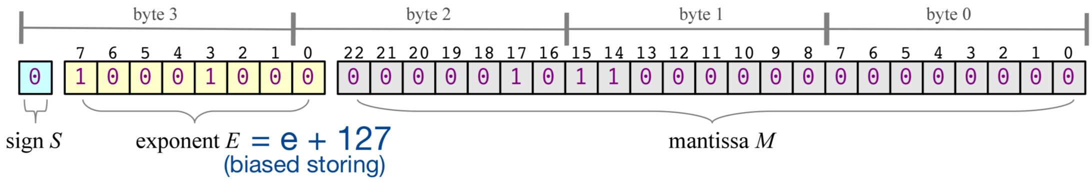


Figure 7: 523 can be written as $1.000001011_2 \cdot 2^9$ so $e = 9, E = e + 127 = 136$. As of the normalization, the leading 1 in the mantissa is implicit. The number is given as a single precision float in big endian.

2.2.2 Only a finite set of floating point numbers can be represented exactly

With 32 bits, 2^{32} states can be encoded (and mind that here e.g. NaN has multiple representations). In any case, the number of exactly representable numbers is finite, above 0 starting at

$$V_{\text{denorm,min}} = \frac{1}{2^p} \cdot 2^{-b+1} \underbrace{\approx}_{\text{single p.}} 1.4 \cdot 10^{-45} \tag{10}$$

with the smallest normalized number being

$$V_{\text{norm,min}} = (1+0) \cdot 2^{-b} \underbrace{\approx}_{\text{single p.}} 1.2 \cdot 10^{-38} \tag{11}$$

and the largest normalized number being

$$V_{\text{norm,max}} = (1 + \frac{2^p - 1}{2^p}) \cdot 2^{E_{\max}-b} \underbrace{\approx}_{\substack{\text{single}\\ \text{p.}}} 3.4 \cdot 10^{38} \tag{12}$$

2.2.3 Machine Precision is finite

The smallest increment in the mantissa, in $1 + \frac{M}{2^p}$, is the **machine precision**

$$\epsilon_{\text{mach}} = \frac{1}{2^p} \underbrace{\approx}_{\substack{\text{single}\\ \text{p.}}} 1.2 \cdot 10^{-7} \tag{13}$$

Consider two floating point numbers $V_1, V_2 > 0$ *next to each other*

$$V_1 = \left(1 + \frac{M}{2^p}\right) \cdot 2^{E-b}, \quad V_2 = \left(1 + \frac{M+1}{2^p}\right) \cdot 2^{E-b} \tag{14}$$

so their relative difference is bound by

$$\frac{V_2 - V_1}{V_2} = \frac{\frac{1}{2^p} \cdot 2^{E-b}}{\left(1 + \frac{M+1}{2^p}\right) \cdot 2^{E-b}} \leq \epsilon_{\text{mach}} \tag{15}$$

2.2.4 Rounding and Pitfalls of Floating Point Arithmetic

In the IEEE standard, results of addition, subtraction, multiplication and division must equal to one where the arithmetic operations are assumed to be exact and then there is rounding to the nearest representable number (computation is done at higher (typically double or higher) precision). Therefore (mind the section before)

$$\text{ralative error } \frac{|x - \hat{x}|}{|x|} \leq \epsilon_{\text{mach}}, \quad \text{number } x, \text{ number on machine } \hat{x} \tag{16}$$

with the common pitfalls

- **Limitation of machine precision:** $a + b = a$ typically for $|b| < \epsilon_{\text{mach}}|a|$, i.e. when b cannot be resolved by the mantissa of a, e.g. $(1 + 0.5\epsilon) - 0.5\epsilon$ in floating point arithmetic yields 0.9999999999999999 not 1.
- **Associativity is not guarantueed:** $(a+b)+c \underbrace{\neq}_{\text{i.A.}} a+(b+c)$, e.g. $(2.3+1e20)-1e20$ yields 0.0 but $2.3 + (1e20 - 1e20)$ yields 2.3

- **Problems of representability:** As e.g. 0.1 is not exactly representable in base $\beta = 2$, $x/10 \neq 0.1 \cdot x$ in general while $x/2.0 = 0.5 \cdot x$ is exact. The compiler may automatically choose the multiplication variant as multiplication is faster than division.
- **Cancellation:** For $x = a - b$ subtractive cancellation causes relative errors already present in $\hat{a}$ and $\hat{b}$ to be (relatively) amplified, when a and b are of similar size. Significant digits are lost and the relative error explodes. Consider e.g. $a = 1.75682, b = 1.75471$ with $\hat{a} = 1.76$ and $\hat{b} = 1.75$. While $\hat{a}, \hat{b}$ have small relative errors (3 digits precision[2] Note that here 0.123 ans 0.127 agree in two significant digits, where one intuitively might say this should be rather 1.), $a - b = 0.00211$ and $\hat{a} - \hat{b} = 0.01$ has a large relative error (no precise digit).
- **NaN and Inf:** All calculations including NaN yield NaN, calculations with inf mostly inf (except of course $1/\text{inf} = 0$, ...)

where we should

- **Rewrite calculations so that errors do not amplify:** For $x = 10^8, y = 10^5, z = -1 - 10^5$ we have $xy + xz = -1.0066 \cdot 10^8$ (problem in resolving the -1 in xz) but $x(y + z) = -1.0 \cdot 10^8$.
- **Compare floats based on a maximum relative error:** Instead of $x == y$ we should use e.g. $|x - y| \leq \epsilon_{\text{mach}} \cdot \max(|x|, |y|)$.
- **As avoid overflow to inf and nan:** E.g. in `float x = 1e20; float y = x * x; float z = y / x` z will be inf.

2.2.5 Rewriting Expressions to Avoid Cancellation I

Consider $f(x) = \frac{1-\cos x}{x^2}$. For $x_e = 10^{-4}$, we have (when we represent 6 figures)

$$c := \cos x_e, \quad \hat{c} = 0.999999, \quad 1 - \hat{c} = 10^{-6}, \quad \text{while } 1 - c \approx 5 \cdot 10^{-9} \tag{18}$$

$1 - c$ has only one significant digit, the relative error is enormously amplified and we get

$$\frac{1 - \hat{c}}{x_e^2} = 100, \quad \text{while in reality for } x \neq 0 : 0 \leq f(x) \leq \frac{1}{2} \tag{19}$$

[2] $\hat{x}$ is said to approximate x to the r-th digit, if the absolute error is at most $\frac{1}{2}$ in the r-th digit, so

$$\text{largest integer s so that } 10^s < |x|, \quad |x - \hat{x}| < 1/2 \cdot 10^{s-r+1} \tag{17}$$

To avoid cancellation we can rewrite using $\cos x = \cos^2 \frac{x}{2} - \sin^2 \frac{x}{2} = 1 - 2\sin^2 \frac{x}{2}$ to

$$f(x) = \frac{1}{2}\left(\frac{\sin \frac{x}{2}}{\frac{x}{2}}\right)^2 \tag{20}$$

A comparison of both versions in python can be found in figure 8, at some point (roughly below 10^{-8}), $\cos x$ is too close to 1 to be resolved by the mantissa, so the total result is 0, above that the magnified error is visible.

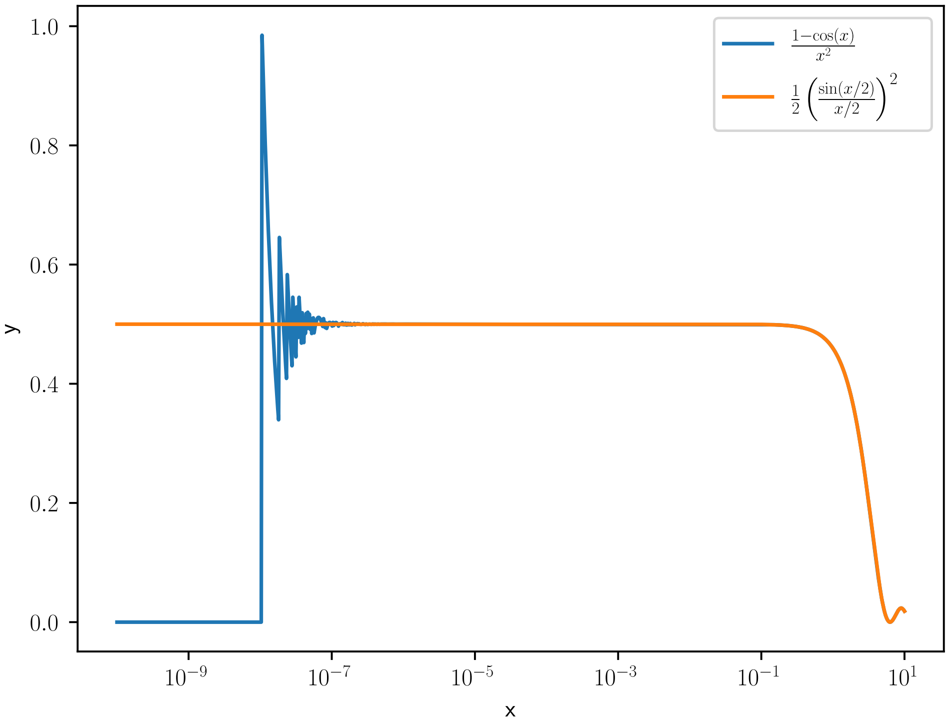


Figure 8: Comparison of the mathematically equivalent expressions $f(x) = \frac{1-\cos x}{x^2}$ and $f(x) = \frac{1}{2}\left(\frac{\sin \frac{x}{2}}{\frac{x}{2}}\right)^2$ in python.

2.2.6 Rewriting Expressions to Avoid Cancellation II

Consider the following expressions for the sample variance of $\{x_i\}_{i=1}^N$

$$\begin{aligned}\text{two-pass formula: } \bar{x} &= \frac{1}{N}\sum_{i=1}^{N-1} x_i, \quad \sigma_N^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i-\bar{x})^2 \\ \text{one-pass formula: } \sigma_N^2 &= \frac{1}{N-1}\left(\sum_{i=1}^{N} x_i^2 - \frac{1}{N}\left(\sum_{i=1}^{N} x_i\right)^2\right) \\ &\text{as } \sigma^2 = E[(x-\bar{x})^2] = E[x^2]-E[x]^2\end{aligned} \tag{21}$$

While for the one-pass formula, we can calculate all necessary sums in one pass through the data, it suffers heavily from cancellation: For $\{10000, 10001, 10002\}$, the two-pass formula in single precision correctly gives 1.0 while the one-pass formula yields 0.0 (cancellation) (there are better one-pass formulas though).

2.2.7 Accumulation of Round-off Errors

Consider the sum

$$\sum_{k=1}^{\infty} k^{-2} = \frac{\pi^2}{6} \tag{22}$$

which we want to approximate by finitely many summands. If we sum the terms just as the formula suggests from large to small, at some point, the small changes will not be resolved anymore - so better sum up from small to large. Summing up in single precision for $N = 10^7$ terms, one gets

$$\begin{aligned}&\text{big to small: } 1.644725323, \quad \text{small to big: } 1.644933939, \\ &\text{exact (till 9th digit): } 1.644934058\end{aligned} \tag{23}$$

As expected the big-to-small summation is too small.

2.2.8 Higher Precision

The above pitfalls are less severe in higher precision. For instance in double precision (64-bit)

$$\begin{aligned}&p = 52(+1) \text{ mantissa bits}, \quad 11 \text{ exponent bits, with } e_{min} = -1022, e_{max} = 1023, \\ &\text{smallest and largest repr. numbers } f_{min} \simeq 2.2 \cdot 10^{-308}, f_{max} \simeq 1.8 \cdot 10^{308}, \\ &|e_{min}| < |e_{max}| \to \frac{1}{f_{min}} < f_{max}\end{aligned} \tag{24}$$

with machine precision $\epsilon_{\text{mach}} \approx 2.2 \cdot 10^{-16}$. There is even quad-double precision (128-bit) (not supported on hardware though and therefore relatively slow as it has to be emulated). Packages for nearly arbitrary precision also exist.

2.3 A more general view on sources of numerical error

In numerical computation, the typical error sources are

- rounding
- data uncertainty
- truncation (of terms in numerical schemata, e.g. in the approximation of a function by its Taylor series)

where we have now discussed rounding errors and their effects to some extent.

2.4 Backward error, forward error and condition number

Consider we approximate $y = f(x)$ as $\hat{y}$ in an arithmetic of limited precision, with $f : \mathbb{R} \to \mathbb{R}$.

- **Forward error:** The absolute or relative error between $\hat{y}$ and y is the forward error (living in the output space)
- **Backward error:** The backward error is the smallest Δx (in the input space) so that $f(x + \Delta x) = \hat{y}$, so the smallest perturbation where the exact function gives our approximate result.

If this Δx is sufficiently small, e.g. as small as the uncertainty in the data in the first place, we speak of backward stability. A weaker formulation is the mixed forward-backward error

$$\hat{y} + \Delta y = f(x + \Delta x), \quad |\Delta y| \leq \epsilon |y|, \quad |\Delta x| \leq \eta |x| \tag{25}$$

In the context of rounding errors, we call the algorithm numerically stable if $\hat{y}$ is almost the right answer for almost the right data (ϵ, η small).

2.4.1 Conditioning

Backward and forward error are connected by the conditioning of a problem, the sensitivity of the solution to perturbations in the data. Assuming $\hat{y} = f(x + \Delta x)$ and f differentiable, we have

$$\hat{y} - y = f(x + \Delta x) - f(x) = f'(x)\Delta x + \mathcal{O}((\Delta x)^2) \tag{26}$$

so the relative error is

$$\frac{\hat{y}-y}{y}=\frac{f'(x)\Delta x}{f(x)}+\mathcal{O}((\Delta x)^2) \tag{27}$$

leading to the relative condition number

$$\kappa(x)\underbrace{=}_{\text{i.A.}}\lim_{\epsilon\to 0^+}\sup_{||\Delta x||\leq\epsilon}\frac{||y-\hat{y}||/||y||}{||\Delta x||/||x||}=\left|\frac{xf'(x)}{f(x)}\right| \tag{28}$$

for small Δx measuring the relative change in the output over a relative change in the input. As a rule of thumb

$$\text{forward error } \lesssim \text{condition number } \cdot \text{ backward error} \tag{29}$$

so ill-conditioned problems can have large forward errors.

Application on Matrices

Consider the linear system $\underline{\underline{A}}\,\underline{y}=\underline{x}, \underline{y}=\underline{\underline{A}}^{-1}\underline{x}, \underline{\hat{y}}=\underline{\underline{A}}^{-1}(\underline{x}+\underline{\Delta x})$. The condition number follows as

$$\begin{aligned}
\kappa(\underline{\underline{A}}) &= \max_{\underline{x},\underline{\Delta x}\neq 0}\frac{||\underline{\underline{A}}^{-1}\underline{x}-\underline{\underline{A}}^{-1}(\underline{x}+\underline{\Delta x})||/||\underline{\underline{A}}^{-1}\underline{x}||}{||\underline{\Delta x}||/||\underline{x}||}\\
&= \max_{\underline{\Delta x}\neq 0}\frac{||\underline{\underline{A}}^{-1}\underline{\Delta x}||}{||\Delta x||}\max_{\underline{x}\neq 0}\frac{||\underline{x}||}{||\underline{\underline{A}}^{-1}\underline{x}||}\\
&= \max_{\underline{\Delta x}\neq 0}\frac{||\underline{\underline{A}}^{-1}\underline{\Delta x}||}{||\underline{\Delta x}||}\max_{\underline{y}\neq 0}\frac{||\underline{\underline{A}}\,\underline{y}||}{||\underline{y}||}\\
&= ||\underline{\underline{A}}^{-1}||\cdot||\underline{\underline{A}}||
\end{aligned} \tag{30}$$

where we used the definition of the matrix norm $||\underline{\underline{A}}||=\max_{\underline{x}\neq 0}\frac{||\underline{\underline{A}}\underline{x}||}{||\underline{x}||}$. For large condition numbers, small perturbations in the input $\underline{x}$ lead to large changes in the solution $\underline{y}$.

Part II

Simulation Methods

The dynamical evolution of physical systems is described using differential equations. Numerical methods for solving differential equations and the rise of computers have allowed for accurate modeling of complex dynamical systems that could hardly be approached by analytical means even under the usage of perturbation theory (compare Moser, 1978).

Oftentimes, we face an initial value problem (IVP) where from initial values from the functions to solve and values for their derivatives as necessary, the evolution is sought to be calculated. In a boundary value problem a differential equation is given together with a set of additional constraints (e.g. Sturm-Liouville problems).

3 Integration of ordinary differential equations

Our aim is solving an ordinary differential equation (ODE) $\partial_t \underline{y} = \underline{f}\left(\underline{y}\right)$ with initial values $\underline{y}(t = t_0) = \underline{y}_0$. Notice that $\underline{f} = \underline{f}(\underline{y}, t)$ can be handled by augmenting $\tilde{\underline{y}} = \begin{pmatrix} \underline{y} \\ t \end{pmatrix}$ and $\tilde{\underline{f}}(\tilde{\underline{y}}) = \begin{pmatrix} \underline{f}(\tilde{\underline{y}}) \\ 1 \end{pmatrix}$.

3.1 Notes on ODEs

3.1.1 Converting to a first order system

Ordinary differential equations only contain derivatives with respect to one variable. Note, however, that higher order derivatives with respect to that variable can occur. We can get to the form $\partial_t \underline{y} = \underline{f}\left(\underline{y}\right)$ by converting to a coupled first order system.

Consider the n-th order ODE

$$\partial_t^n y(t) = f\left(y(t), \partial_t y(t), \ldots, \partial_t^{n-1} y(t), t\right), \quad f : U \subset \mathbb{R} \times \mathbb{K}^n \to \mathbb{K} \tag{31}$$

for instance a pendulum with damping

$$\partial_t^2 \phi = -\omega_0^2 \sin\phi - \gamma \partial_t \phi, \quad \gamma, \omega_0 \in \mathbb{R} \tag{32}$$

Now we define the variables

$$u_m = \partial_t^m y(t), \quad m \in \{0, \ldots, n-1\} \tag{33}$$

leading to the coupled first order system

$$\partial_t \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-2} \\ u_n \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ f\left(t, u_0, u_1, \ldots, u_{n-1}\right) \end{pmatrix} \rightarrow \partial_t \underline{u} = \underline{f}\left(t, \underline{u}\right) \tag{34}$$

Using ϕ for the angle and $\omega = \partial_t \phi$ for the angular velocity, we can write the pendulum as

$$\partial_t \begin{pmatrix} \phi \\ \omega \end{pmatrix} = \begin{pmatrix} \omega \\ -\omega_0^2 \sin\phi - \gamma\omega \end{pmatrix} \tag{35}$$

3.1.2 Existence and uniqueness of an ODE solution for an initial value problem - Picard-Lindelöf and Lipschitz condition

For the initial value problem $\partial_t \underline{y} = \underline{f}\left(\underline{y}\right), \underline{y}(t_0) = \underline{y}_0$ to have a unique solution in the vicinity of $(\underline{y}_0, t_0)$, i.e. for the change around that point, to uniquely determine the development, this change must be *well-behaved*, $\underline{f}$ must be *Lipschitz-continuous*.

$$\forall (\underline{y}, t), (\underline{z}, t) \text{ in the vicinity of } (\underline{y}_0, t_0) : ||\underline{f}(\underline{y}, t) - \underline{f}(\underline{z}, t)|| \leq \lambda ||\underline{y} - \underline{z}|| \tag{36}$$

with $\lambda > 0$ and $|| \cdot ||$ being an arbitrary vector norm. The slope of the line connecting two close-by evaluations of $\underline{f}$ must be bounded by λ. This is guaranteed for $\underline{f}$ being continuous and sufficiently often differentiable with bounded derivatives and more so $\underline{f}$ analytic.

3.2 Introduction of Numerical Integration at the hand of the two-body problem

Our aim is computationally modelling the interaction of two-bodies. This lends itself well as an example, as stepping the system forward in time is easy to imagine visually, an analytic solution exists to which we might compare numerical solution and it guides us to the problem of conserved quantities and symplectic integrators.

3.2.1 The two-body problem

For the two-body problem (illustrated in figure 9) the equations of motion are

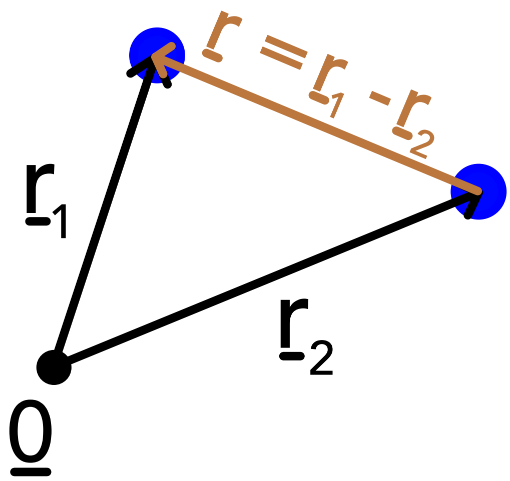


Figure 9: Illustration of the two-body problem.

$$\begin{aligned} m_1 \partial_t^2 \underline{r}_1 &= -G \frac{m_1 m_2}{|\underline{r}|^3} \underline{r} \\ m_2 \partial_t^2 \underline{r}_2 &= +G \frac{m_1 m_2}{|\underline{r}|^3} \underline{r} \end{aligned} \tag{37}$$

for $\underline{r} = \underline{r}_1 - \underline{r}_2$. Subtracting both yields

$$\partial_t^2 \underline{r} = -G \frac{M}{|\underline{r}|^3} \underline{r} \tag{38}$$

with $M = m_1 + m_2$. Which is equivalent to the equation of motion of a single body of mass $\mu = \frac{m_1 m_2}{M}$ in a potential $U(r) = -G\frac{m_1 m_2}{r} = -G\frac{M\mu}{r}$.

We can write this as the first order system

$$\partial_t \begin{pmatrix} \underline{r} \\ \underline{v} \end{pmatrix} = \begin{pmatrix} \underline{v} \\ -G\frac{M}{|\underline{r}|^3}\underline{r} \end{pmatrix} \tag{39}$$

3.2.2 Integrals of Motion

The following quantities are conserved along the trajectories of m_1 and m_2 and are therefore useful sanity checks for simulations.

- Total energy

$$E = T + U = \frac{\mu}{2} v^2 - \frac{GM}{r}\mu \tag{40}$$

- Angular momentum (perpendicular to the orbital plane)

$$\underline{L} = \underline{r} \times \underline{p} = \underline{r} \times \mu \underline{v} \tag{41}$$

- Laplace-Runge-Lenz vector (here in its dimensionless form, the eccentricity vector)

$$\underline{e} = \frac{\underline{v} \times \underline{j}}{GM} - \underline{\hat{e}}_r, \quad \text{specific angular momentum } \underline{j} = \frac{\underline{L}}{\mu} \text{ eccentricity } e = ||\underline{e}|| \tag{42}$$

Note: The 1-body Kepler problem has 6 degrees of freedom (phase-space coordinates), of which one cannot be conserved, as nothing should be able to tell us the initial time of our motion. Therefore, only 5 quantities can be conserved and the Laplace-Runge-Lenz vector indeed only adds 1 more conserved degree of freedom (taking E and $\underline{L}$ as primary conserved quantities, $\underline{e}$ only has one degree of freedom).

Additional notes on the Laplace-Runge Lenz vector

The Lenz vector is conserved in all $\frac{1}{r}$-potentials, like the gravitational or Coulomb potential, for instance in the Hydrogen atom (but not for multi-electron atoms). Kepler-orbits are conic sections and the Laplace-Runge-Lenz vector is illustrated in figure 10.

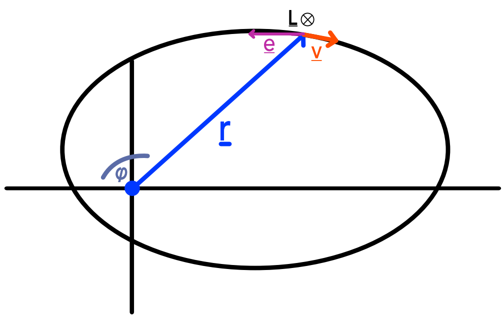


Figure 10: Illustration of the Laplace-Runge-Lenz vector.

From our pictorial evidence, we see that $\underline{e}$ is points along the semi-major axis. Note here we have drawn that $\underline{r} = \underline{r}_1 - \underline{r}_2$ follows a conic section. Likewise m_1 and m_2 move on conic sections with respect to the center of mass, $\underline{0} \underbrace{=}_{\text{o.b.d.A.}} \frac{1}{M}\left(m_1\underline{r}_1 + m_2\underline{r}_2\right)$ leading to $\underline{r}_1 = \frac{m_2}{M}\underline{r}$ and $\underline{r}_2 = -\frac{m_1}{M}\underline{r}$.

3.2.3 Kepler Orbits are Conic Sections

Depending on the total energy E, we have

- $E < 0 \rightarrow$ (closed) elliptic orbit

- $E = 0 \rightarrow$ parabolic orbit
- $E > 0 \rightarrow$ hyperbolic orbit

This dependence of the orbit form on the energy can be seen from writing

$$E = \frac{\mu}{2}(\partial_t \underline{r})^2 + U(r), \quad U(r) = -\frac{\alpha}{r}\mu, \qquad \text{here } \alpha = GM \tag{43}$$

and using polar coordinates as the movement takes place on a planar surface

$$(\partial_t \underline{r})^2 = (\partial_t r)^2 + r^2(\partial_t \phi)^2 \tag{44}$$

with $\partial_t \phi$ expressed via the conserved angular momentum

$$\text{const.} \ = l = I\omega = \mu r^2 \partial_t \phi \quad \Rightarrow \quad \partial_t \phi = \frac{l}{\mu r^2} \tag{45}$$

so

$$\begin{aligned} E &= \frac{\mu}{2}(\partial_t \underline{r})^2 + U(r) \\ &= \frac{\mu}{2}(\partial_t r)^2 + \frac{\mu}{2} r^2 (\partial_t \phi)^2 + U(r) \\ &= \frac{\mu}{2}(\partial_t r)^2 + \underbrace{\frac{l^2}{2\mu r^2} + U(r)}_{U_{eff}} \end{aligned} \tag{46}$$

Note here that we have expressed the total energy as the sum of a kinetic part stemming from a change in distance between the two bodies (which we have already related to the individual positions) and an effective potential U_{eff}. Where the vertical line of constant energy intersects the effective potential, $\partial_t r$ must be zero so such a point must be a point of reversal of movement (see figure 11 and 12).

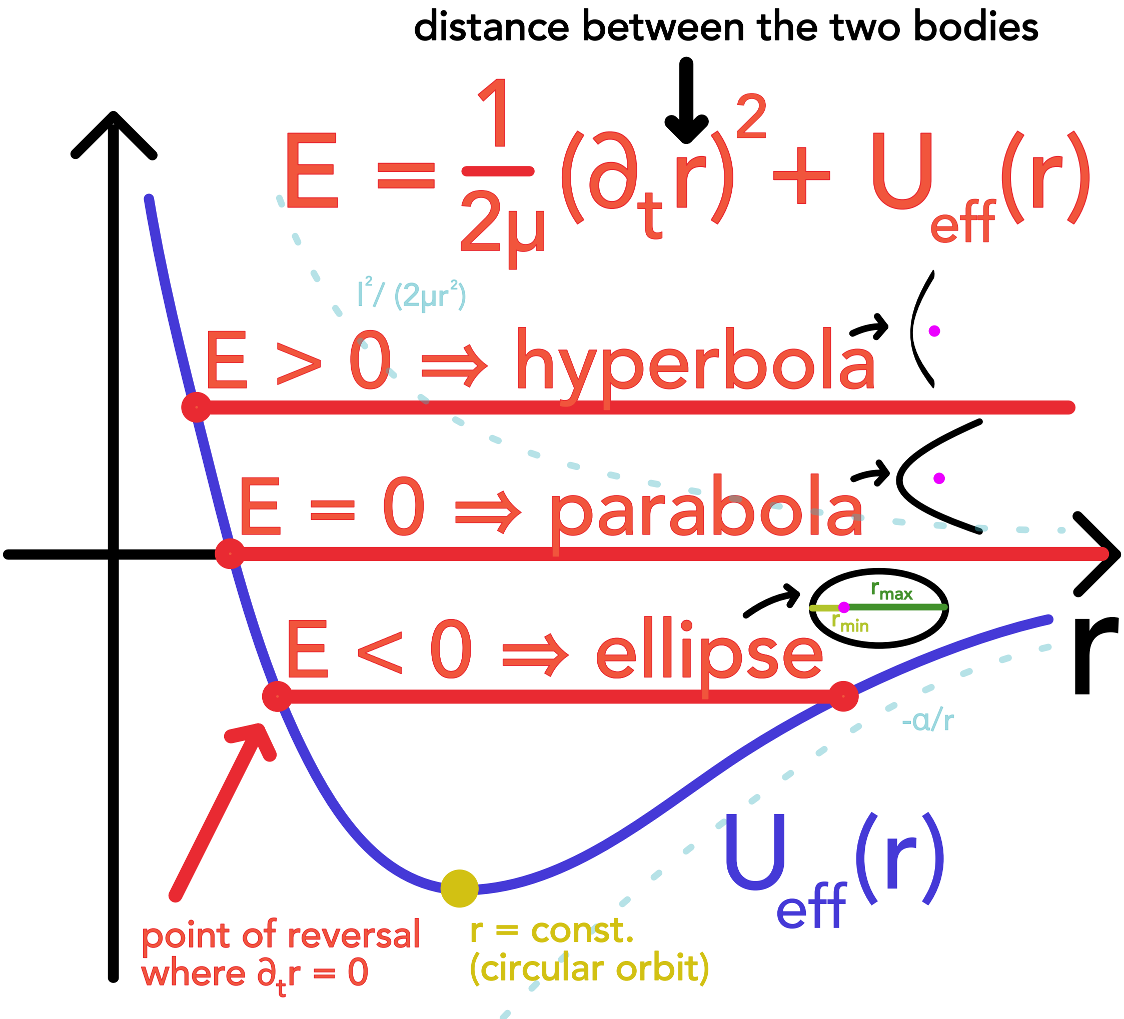


Figure 11: Illustration of the effective potential U_{eff} and the resulting orbits.

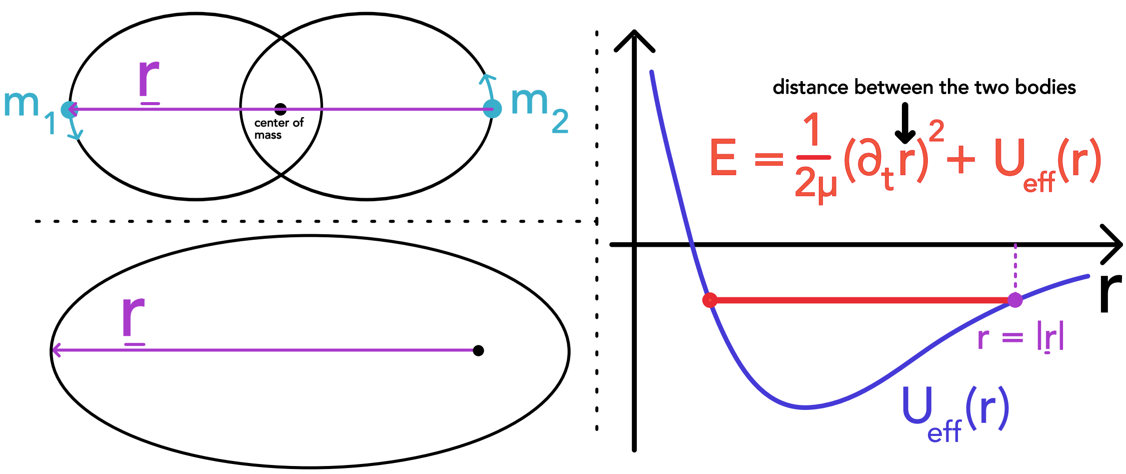


Figure 12: Connection between the different views on the two-body problem.

3.2.4 Connection of the Runge-Lenz vector to the eccentricity of a conic section

From multiplying $\underline{e}$ with $\underline{r}$ we obtain

$$\underline{e}\cdot\underline{r} = er\cos\varphi = \frac{(\underline{v}\times\underline{j})\cdot\underline{r}}{GM} - r \underset{\underline{a}\cdot(\underline{b}\times\underline{c})=\underline{b}\cdot(\underline{c}\times\underline{a})=\underline{c}\cdot(\underline{a}\times\underline{b})}{=} \frac{(\underline{r}\times\underline{v})\cdot\underline{j}}{GM} - r = \frac{j^2}{GM} - r$$
$$\rightarrow r(\varphi) = \frac{j^2/GM}{1+\epsilon\cos\varphi} \tag{47}$$

3.2.5 Rescaling to Dimensionless variables

While the relative precision with which a number is stored on a computer is $\sim$ the machine precision, so independent of magnitude, we rescale our variables so that they predominantly fall into the range $[-1,1]$ so that different variables are on the same scale (so also their absolute precisions) (also making the problem statement more general).

$$\underline{r} \quad \rightarrow \quad \underline{s} := \frac{\underline{r}}{R_0}, \quad \text{characteristic length } R_0 \text{ e.g. initial separation} \tag{48}$$

$$\underline{v} \quad \rightarrow \quad \underline{w} := \frac{\underline{v}}{v_0}, \quad \text{characteristic velocity } v_0 = \left(\frac{GM}{R_0}\right)^{1/2} \tag{49}$$

v_0 is the velocity, a body circling one with mass M at distance R_0 would have ($F_{zp} = F_G$).

$$t \quad \rightarrow \quad \tau := \frac{t}{T_0}, \quad \text{characteristic time } T_0 = \frac{R_0}{v_0} = \left(\frac{R_0^3}{GM}\right)^{1/2} \tag{50}$$

With this we can write the equation of motion as

$$\frac{d\underline{s}}{d\tau} = \underline{w}, \quad \frac{d\underline{w}}{d\tau} = -\frac{\underline{s}}{|\underline{s}|^3} \tag{51}$$

3.2.6 Solving the two-body problem using explicit (aka forward) Euler

Let us discretize the derivatives with a simple difference quotient, where we probe the current slope by comparing the current position to the one a small time-step in the past or future.

$$\frac{d\underline{s}^{(n)}}{d\tau} = \frac{\underline{s}^{(n)} - \underline{s}^{(n-1)}}{h} + \mathcal{O}(h)(\text{backwards}) \quad \text{or} \quad \frac{d\underline{s}^{(n-1)}}{d\tau} = \frac{\underline{s}^{(n)} - \underline{s}^{(n-1)}}{h} + \mathcal{O}(h)(\text{forward}) \tag{52}$$

where $h = \tau^{(n)} - \tau^{(n-1)}$ is the step-size and the *forward* formulation gives an explicit scheme for $\underline{s}_n$ (only depending on already known values)

$$\begin{aligned} \underline{s}^{(n)} &= \underline{s}^{(n-1)} + h\frac{d\underline{s}^{(n-1)}}{d\tau} + \mathcal{O}(h^2) = \underline{s}^{(n-1)} + h\underline{w}^{(n-1)} + \mathcal{O}(h^2) \\ \underline{w}^{(n)} &= \underline{w}^{(n-1)} + h\frac{d\underline{w}^{(n-1)}}{d\tau} + \mathcal{O}(h^2) = \underline{w}^{(n-1)} - h\frac{\underline{s}^{(n-1)}}{|\underline{s}^{(n-1)}|^3} + \mathcal{O}(h^2) \end{aligned} \tag{53}$$

(explicit Euler)

and the *backward* formulation gives an implicit scheme for $\underline{s}_n$ (*implicit* as depending on the yet unknown $\underline{w}^{(n)}$).

$$\begin{aligned} \underline{s}^{(n)} &= \underline{s}^{(n-1)} + h\frac{d\underline{s}^{(n)}}{d\tau} + \mathcal{O}(h^2) = \underline{s}^{(n-1)} + h\underline{w}^{(n)} + \mathcal{O}(h^2) \\ \underline{w}^{(n)} &= \underline{w}^{(n-1)} + h\frac{d\underline{w}^{(n)}}{d\tau} + \mathcal{O}(h^2) = \underline{w}^{(n-1)} - h\frac{\underline{s}^{(n)}}{|\underline{s}^{(n)}|^3} + \mathcal{O}(h^2) \end{aligned} \tag{54}$$

This is also very clear just from first order Taylor expansion.

3.2.7 Probing the accuracy of an integration scheme - energy error of explicit Euler

We probe the accuracy, by checking on the conserved quantities (now dimensionless)

$$\begin{gathered} \text{total energy } E^{(n)} = \frac{\left(\underline{w}^{(n)}\right)^2}{2} + \frac{1}{s^{(n)}}, \quad \text{angular momentum } \underline{j}^{(n)} = \underline{s}^{(n)} \times \underline{w}^{(n)} \\ \text{Laplace - Runge - Lenz vector } \underline{e}^{(n)} = \underline{w}^{(n)} \times \left(\underline{s}^{(n)} \times \underline{w}^{(n)}\right) - \underline{s}^{(n)} \end{gathered} \tag{55}$$

Wanted behavior: In a good integration scheme, the truncation errors (from the Taylor expansion) and rounding errors should be small or at least not accumulate without bound.

We calculate relative errors with respect to the initial values, e.g.

$$\epsilon^{(n)}(h) = \frac{|E^{(n)} - E^{(0)}|}{|E^{(0)}|}$$

Rough error estimation for explicit Euler: Each step has an error of $\mathcal{O}(h^2)$, an orbit takes $\sim \frac{T_0}{\Delta t} = \frac{1}{h}$ steps so we expect an error of $\mathcal{O}(h)$ per orbit, more on the problem of applying *non-symplectic* schemes onto *symplectic* problems follow later.

3.3 Explicit Euler and it's shortcomings

The simplest method for solving an ODE is the Explicit Euler method

$$\underline{y}^{(n+1)} = \underline{y}^{(n)} + \underline{f}\left(\underline{y}^{(n)}\right)\Delta t, \quad \text{where} \quad \underline{y}^{(0)} = \underline{y}_0$$

which is explicit as the computation of $\underline{y}^{(n+1)}$ only depends on already known states.

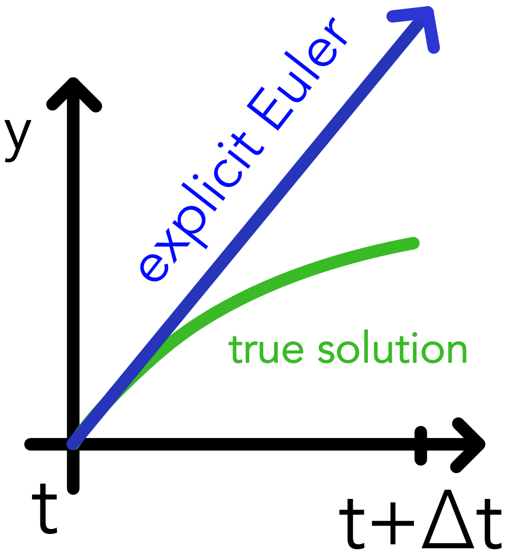


Figure 13: Illustration of one time step in the Explicit Euler scheme.

As illustrated in Figure 13 in every step we step forward along the current derivative $\underline{f}\left(\underline{y}^{(n)}\right)$.

3.3.1 Explicit Euler is only first order accurate | truncation error

A simple error approximation follows from Taylor expansion

$$\underline{y}(t + \Delta t) = \underline{y}(t) + \Delta t \underline{f}(t) + \mathcal{O}_s(\Delta t^2)$$

In each step we make an error $\mathcal{O}_s(\Delta t^2)$ so over some timespan T where we need $N_S = \frac{T}{\Delta t}$ steps we accumulate the error $N_S \mathcal{O}_S(\Delta t^2) = \mathcal{O}_T(\Delta t)$. We therefore call Explicit Euler first order accurate.

Note: For a global error scaling with $\mathcal{O}_T(\Delta t^n)$ (n-th order accurate scheme), the local truncation error (of the Taylor expansion) must be $\mathcal{O}_s(\Delta t^{n+1})$.

3.3.2 Explicit Euler has stability issues

Stability analysis is a broad field, and the interested reader can find details in chapter IV.3 of Hairer and Wanner, 1996. For now, consider the ODE $\partial_t y = \alpha y, Re(\alpha) < 0, y(0) = y_0$

with the solution $y(t) = y_0 e^{\alpha y}$.

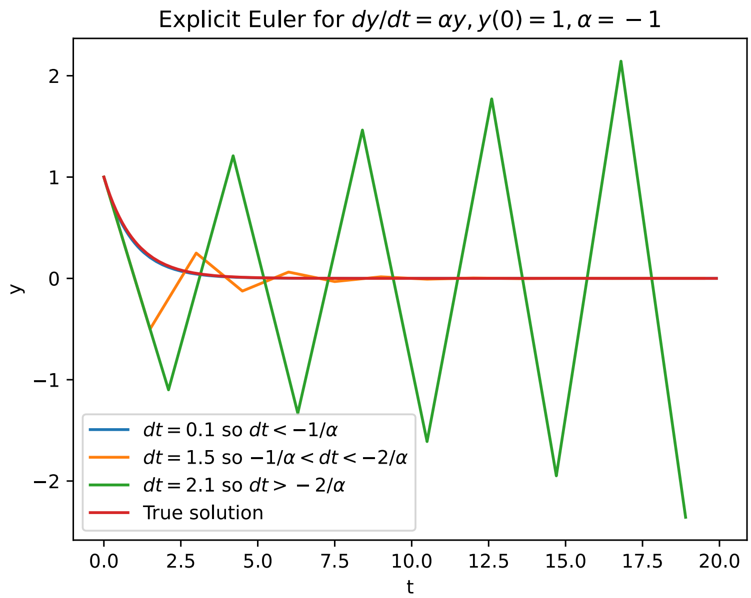


Figure 14: Linear stability of the Explicit Euler scheme.

The results of applying Explicit Euler for different step sizes Δt are shown in figure 14. At a small step size the correct solution is obtained, for a larger step size the numerical solution becomes oscillatory and for even larger step sizes it diverges. We can quantitatively explain this behavior by looking at the Euler steps

$$\begin{aligned} y^{(n+1)} &= y^{(n)} + \alpha y^{(n)} \Delta t \\ &= y^{(n)}(1 + \alpha \Delta t) \\ &= y^{(0)}(1 + \alpha \Delta t)^{n+1} \end{aligned}$$

- $\Delta t < -\frac{1}{\alpha} \rightarrow$ we observe monotonous decrease (ok)
- $-\frac{1}{\alpha} < \Delta t < -\frac{2}{\alpha} \rightarrow$ oscillation (regarding the sign) but still decrease in the absolute value (problematic)
- $-\frac{2}{\alpha} < \Delta t \rightarrow$ an increasing, oscillating solution (very bad)

The growth factor $R(\alpha \Delta t) = 1 + \alpha \Delta t$ in $y^{(n+1)} = R(\alpha \Delta t) y^{(n)}$ is called stability function and

$$\mathcal{D} := \{z \in \mathcal{C} : |R(z)| \leq 1\} \quad \text{so} \quad D_{Euler} = \{z = \alpha \Delta t \in \mathcal{C} : |1 + z| \leq 1\}$$

is called region of absolute stability or linear stability domain. D_{Euler} is a finite region of absolute stability in form of a circle on the left of the complex plane (see figure 15).

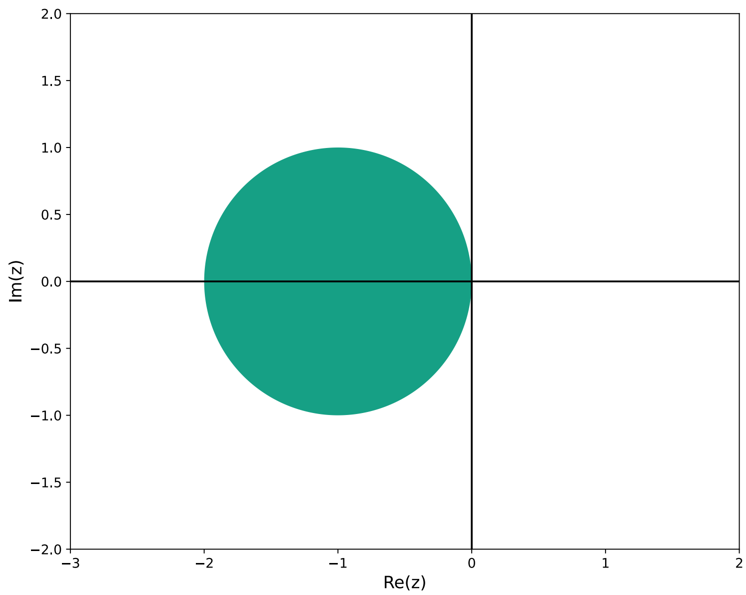


Figure 15: Region of absolute stability of the Explicit Euler method.

Problem: While in this example the stability constraint is easy to fulfill (we get a good solution for a reasonably large step-size), in problems with different timescales, with explicit Euler we must resolve the fastest one, even if its completely negligible (*stiff problems*).

More on stability, A-stable, L-stable, ...

3.4 Introduction of the Problem of Stiffness and Implicit Euler to the help

3.4.1 Introducing stiffness at the hand of a simple example

Consider the following ODE system (following Press et al., 2007, chapter 17.5)

$$\partial_t y_1 = 998 y_1 + 1998 y_2$$
$$\partial_t y_2 = -999 y_1 - 1999 y_2$$

with initial conditions $y_1(0) = 1$ and $y_2(0) = 0$. The system can be represented in matrix form as

$$\partial_t \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \underline{\underline{A}} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \quad \underline{\underline{A}} = \begin{pmatrix} 998 & 1998 \\ -999 & -1999 \end{pmatrix}$$

The eigenvalues of $\underline{\underline{A}}$ are $\lambda_1 = -1$ and $\lambda_2 = -1000$. The eigenvectors are $\underline{e}_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$ and $\underline{e}_2 = \begin{pmatrix} 2 \\ -1 \end{pmatrix}$. The solution of the system is then

$$\begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} = \exp\left(\underline{\underline{A}}t\right)\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 2 \\ -1 \end{pmatrix}\exp(-1t) + \begin{pmatrix} -1 \\ 1 \end{pmatrix}\exp(-1000t)$$

Let us now apply the Explicit Euler method to this system for different time-steps Δt. The result is shown in figure 16.

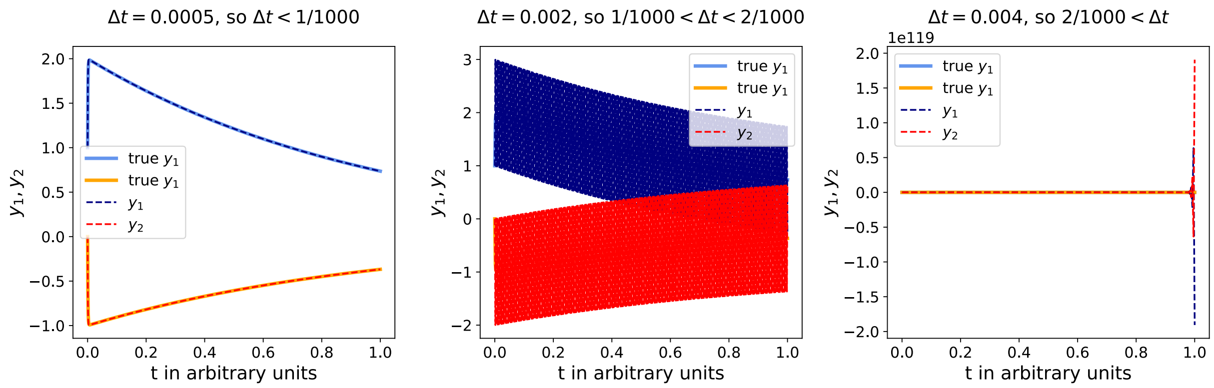


Figure 16: Numerical solution to the linear system $\partial_t \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \underline{\underline{A}} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$ with $\underline{\underline{A}} = \begin{pmatrix} 998 & 1998 \\ -999 & -1999 \end{pmatrix}$ and $y_1(0) = 1,\ y_2(0) = 0$ using the Explicit Euler method for different time-steps Δt. The left panel shows the solution for $\Delta t = 0.0005$, the central one for $\Delta t = 0.002$ and the right one for $\Delta t = 0.004$.

Let us think back to the linear stability analysis of the Explicit Euler scheme for $\partial_t y = \alpha y, Re(\alpha) < 0, y(0) = y_0$. We had obtained

- $\Delta t < -\frac{1}{\alpha} \rightarrow$ we observe monotonous decrease (ok)
- $-\frac{1}{\alpha} < \Delta t < -\frac{2}{\alpha} \rightarrow$ oscillation (regarding the sign) but still decrease in the absolute value (problematic)
- $-\frac{2}{\alpha} < \Delta t \rightarrow$ an increasing, oscillating solution (very bad)

The same result holds in principle for our linear system - but with α replaced by the eigenvalue of largest magnitude of $\underline{\underline{A}}$, here $\lambda_2 = -1000$ (for the proof see Press et al., 2007, chapter 17.5).

As we move away from the origin, the fastest decreasing term $\propto \exp(-\lambda_2 t)$ in the true solution is completely negligible. However, in the explicit scheme it still sets the timescale that has to be resolved for a stable solution.

In the setting of $\partial_t y = \alpha y, Re(\alpha) < 0$ the stability constraint for Δt is not too problematic because the resulting step-size is reasonable compared to the timescale of the problem. In the case of an ODE with different timescales in the solution, however, we are often interested in the timescale of the slowest processes but in the explicit scheme we still need to resolve the fastest timescale which quickly becomes infeasable. This is the problem of stiffness and can - in such a linear setting with all negative eigenvalues of $\underline{\underline{A}}$ - be characterized by the stiffness ratio

$$\text{stiffness ratio} := \frac{\max_{\text{eigenvalues } \lambda_i \text{ of } \underline{\underline{A}}} |\text{Re } \lambda_i|}{\min_{\text{eigenvalues } \lambda_i \text{ of } \underline{\underline{A}}} |\text{Re } \lambda_i|} = \frac{\lambda_2}{\lambda_1} = 1000$$

A large stiffness ratio indicates that an explicit scheme like the Explicit Euler method would be very inefficient for following the slowest process.

3.4.2 A *definition* of stiffness

As discussed in Lambert, 1991 a hard mathematical definition of stiffness is difficult and we therefore resort to the broad practical definition (Lambert, 1991, chapter 6)

> »If a numerical method with a finite region of absolute stability, applied to a system with any initial conditions, is forced to use in a certain interval of integration a step length which is excessively small in relation to the smoothness of the exact solution in that interval, then the system is said to be stiff in that interval.«

An example for a numerical method with a finite region of absolute stability is the Explicit Euler method (see figure 15). In the example above, in spite of the fact that the solution is very smooth and the term $\propto \exp(-\lambda_2 t)$ is quickly negligible, we have to use excessively small steps.

3.4.3 Implicit Euler to the help

At the core of dealing with stiffness are implicit methods, the simplest representative being Implicit Euler.

An Implicit Euler step for solving $\partial_t \underline{y} = \underline{f}\left(\underline{y}\right)$ is given by

$$\underline{y}^{(n+1)} = \underline{y}^{(n)} + \underline{f}\left(\underline{y}^{(n+1)}\right) \Delta t \quad \text{where} \quad \underline{y}^{(0)} = \underline{y}_0$$

which is an implicit equation as $\underline{f}$ is evaluated at the new time step $\underline{y}^{(n+1)}$.

Intuition behind implicit Euler: We can write the implicit Euler step as $\underline{y}^{(n+1)} - \underline{f}\left(\underline{y}^{(n+1)}\right)\Delta t = \underline{y}^{(n)}$, so which is the point where when I sit on it and shoot back with the corresponding slope, I get back to where I am coming from. This is illustrated in figure 17.

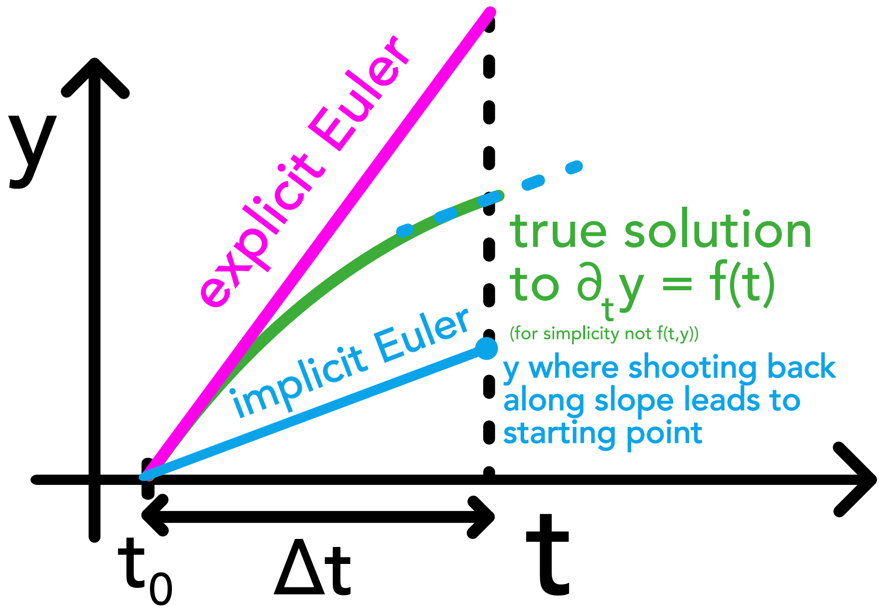


Figure 17: Illustration of the implicit Euler step.

Note: Implicit Euler is often referred to as backward Euler and the explicit Euler as forward Euler.

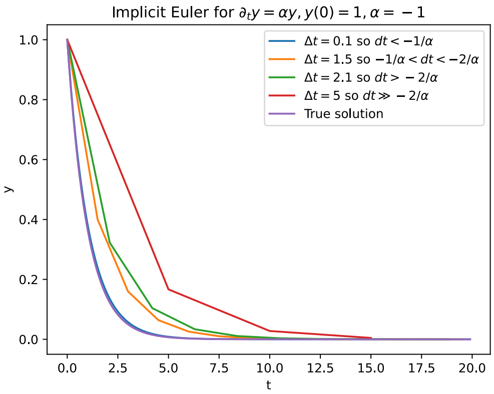
Problem: Note that implicit Euler is also a first order accurate scheme.

3.4.4 Region of absolute stability of the Implicit Euler method

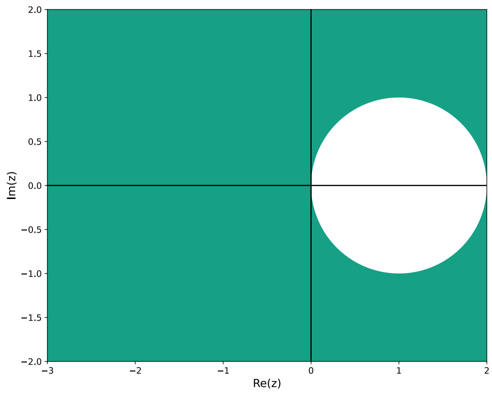
As for the Explicit Euler method, we perform a linear stability analysis of the Implicit Euler method for $\partial_t y = \alpha y, Re(\alpha) < 0, y(0) = y_0$. We obtain

$$y^{(n+1)} = y^{(n)} + \alpha y^{(n+1)}\Delta t \quad \Rightarrow \quad y^{(n+1)} = \frac{1}{1-\alpha\Delta t}y^{(n)}$$

which decreases for any $\Delta t > 0$ (illustrated in figure 18a). For large time-steps, the result is inaccurate (Implicit Euler is a first order scheme) but the solution remains stable. As of the stability function $R(z) = \frac{1}{1-z}$ the region of absolute stability is given by



(a) Linear stability of Implicit Euler.



(b) Region of absolute stability of the Implicit Euler method (shaded in green).

Figure 18: Stability of the Implicit Euler scheme.

$$\mathcal{D}_{\text{implicit euler}} = \{z \in \mathbb{C} \mid |R(z)| < 1\} = \{z \in \mathbb{C} \mid |1 - z| > 1\}$$

which is illustrated in figure 18b. The whole left half plane is included in the region of absolute stability and the method is therefore unconditionally stable.

3.4.5 Implicit Euler for stiff linear ODEs

As Implicit Euler is unconditionally stable, the fast oscillating terms resulting from

$$\partial_t \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \underline{\underline{A}} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \quad \underline{\underline{A}} = \begin{pmatrix} 998 & 1998 \\ -999 & -1999 \end{pmatrix}$$

with initial conditions $y_1(0) = 1$ and $y_2(0) = 0$ are no problem as illustrated in figure 19, where in spite of the relatively large time-step a good approximation of the solution is obtained.

The implicit step for such a linear system $\partial_t \underline{y} = \underline{\underline{A}}\underline{y}$ is

$$\underline{y}^{(n+1)} = \underline{y}^{(n)} + \underline{\underline{A}}\underline{y}^{(n+1)} \Delta t \quad \Rightarrow \quad \left(\underline{\underline{1}} - \underline{\underline{A}} \Delta t\right) \underline{y}^{(n+1)} = \underline{y}^{(n)}$$

which means that to make a step we have to solve a linear system which is usually done by matrix decomposition (like LU decomposition).

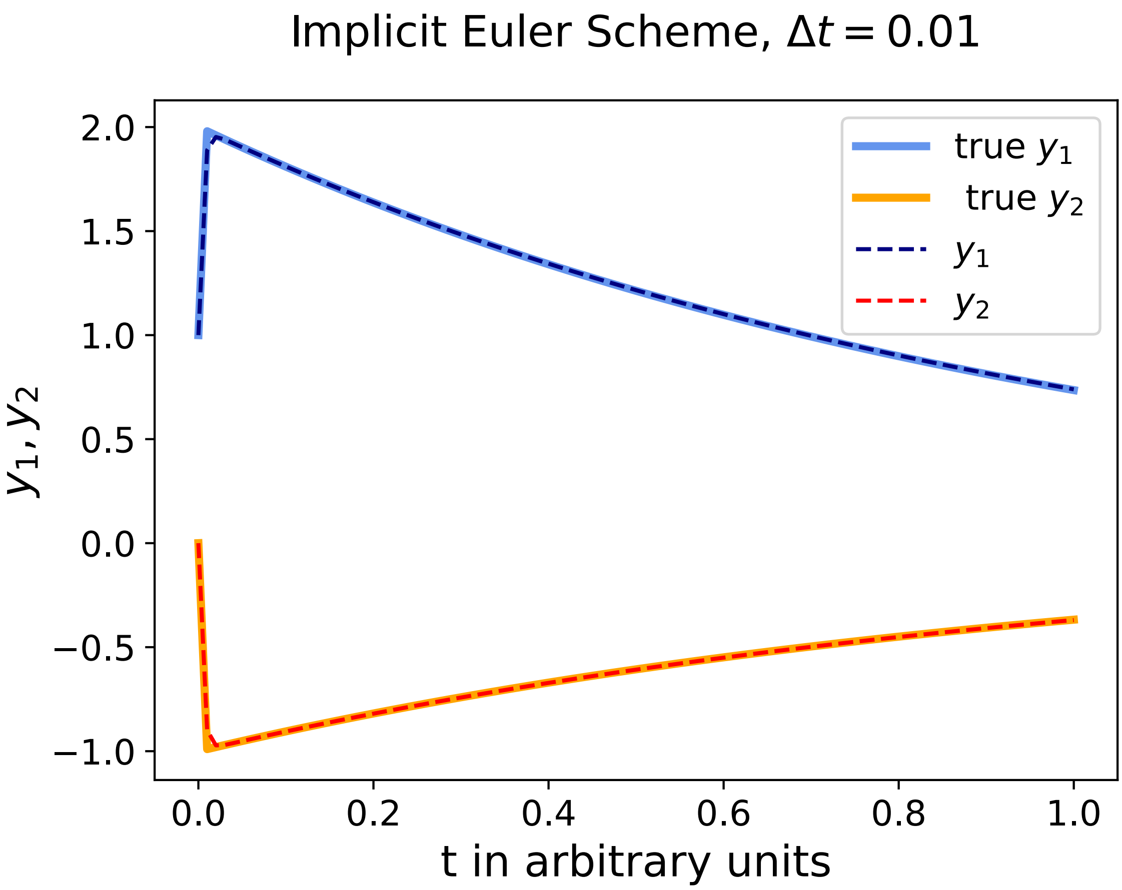


Figure 19: The same problem as in figure 16 is now approached using the Implicit Euler method and a relatively large time-step of $\Delta t = 0.01$.

3.4.6 But how can we approach non-linear ODEs using the Implicit Euler method?

To perform an implicit step

$$\underline{y}^{(n+1)} = \underline{y}^{(n)} + \underline{f}\left(\underline{y}^{(n+1)}\right)\Delta t$$

for a non-linear system $\partial_t \underline{y} = \underline{f}\left(\underline{y}\right)$ like the Davis-Skodje equation

$$\begin{aligned}\dot{y}_1(t) &= -y_1(t)\\ \dot{y}_2(t) &= -\gamma y_2(t) + \frac{(\gamma-1)y_1(t) + \gamma y_1^2(t)}{\left(1+y_1(t)\right)^2}\end{aligned}$$

where γ is a measure for the stiffness (see Heiter, 2012, chapter 2.4) we reformulate the implicit step as a root-finding problem

$$\underline{0} = \underline{y}^{(n+1)} - \underline{y}^{(n)} - \Delta t \underline{f}\left(\underline{y}^{(n+1)}\right), \quad \underline{g}\left(\underline{\xi}\right) := \underline{\xi} - \underline{y}^{(n)} - \Delta t \underline{f}\left(\underline{\xi}\right)$$
$$\rightarrow \underline{0} = \underline{g}\left(\underline{\xi}\right) \Leftrightarrow \underline{\xi} = \underline{y}^{(n+1)}$$

where each of those time-steps is solved using Newton's method (or quasi-Newton)

$$\underline{\xi}_{k+1} = \underline{\xi}_k - \underline{\underline{J}}_g^{-1}\left(\underline{\xi}_k\right) \underline{g}\left(\underline{\xi}_k\right), \quad \underline{\underline{J}}_g = \underline{\underline{1}} - \Delta t \gamma \left(\underline{\xi}_k\right) \underline{\underline{J}}_f$$

$$\underline{\xi}_0 = \underline{y}^{(n)}, \quad \underline{\xi}_m \rightarrow \underline{y}^{(n+1)} \quad \text{for} \quad m \rightarrow \infty$$

where $\underline{\underline{J}}_f$ is the Jacobian of $\underline{f}$. In Quasi-Newton the Jacobian is only recalculated once per time-step in the Euler method

$$\underline{\xi}_{k+1} = \underline{\xi}_k - \underline{\underline{J}}_g^{-1}\left(\underline{\xi}_0\right) \underline{g}\left(\underline{\xi}_k\right)$$

For the Davis-Skodje problem mentioned above some Implicit Euler steps are drawn into the stream plot of the equation in figure 20. Here, one can also see the intuition behind Implicit Euler steps: One searches a point where the derivative is such that shooting back with this slope leads back to the point we are coming from, as

$$\underline{y}^{(n)} = \underline{y}^{(n+1)} - \underline{f}\left(\underline{y}^{(n+1)}\right) \Delta t$$

The steps of the Newton iteration done for each Implicit Euler step can most intuitively be understood in the formulation as the linear equation

$$\underline{b} := \underline{g}\left(\underline{\xi}_k\right) = \underline{\underline{J}}_g \left(\underline{\xi}_k - \underline{\xi}_{k+1}\right) = \underline{\underline{J}}_g \underline{a}, \quad \underline{a} := \underline{\xi}_k - \underline{\xi}_{k+1}$$

which is also the equation solved on the computer using matrix decomposition. $\underline{\underline{J}}_g \underline{a}$ is the directional derivative of $\underline{g}$ in the direction of $\underline{a}$ and in a step of the Newton iteration we search for a step $\underline{a}$ that gets us from $\underline{0}$ to $\underline{b}$ in other words $\underline{\xi}_{k+1} = \underline{\xi}_k - \underline{a}$.

Problem: While the Implicit Euler method is unconditionally stable, performing the implicit step for non-linear ODEs requires solving a non-linear equation with some root-finding algorithm, which can be even more costly than doing small explicit steps if no proper care (e.g. smart forward differentiation in the root finding) is taken.

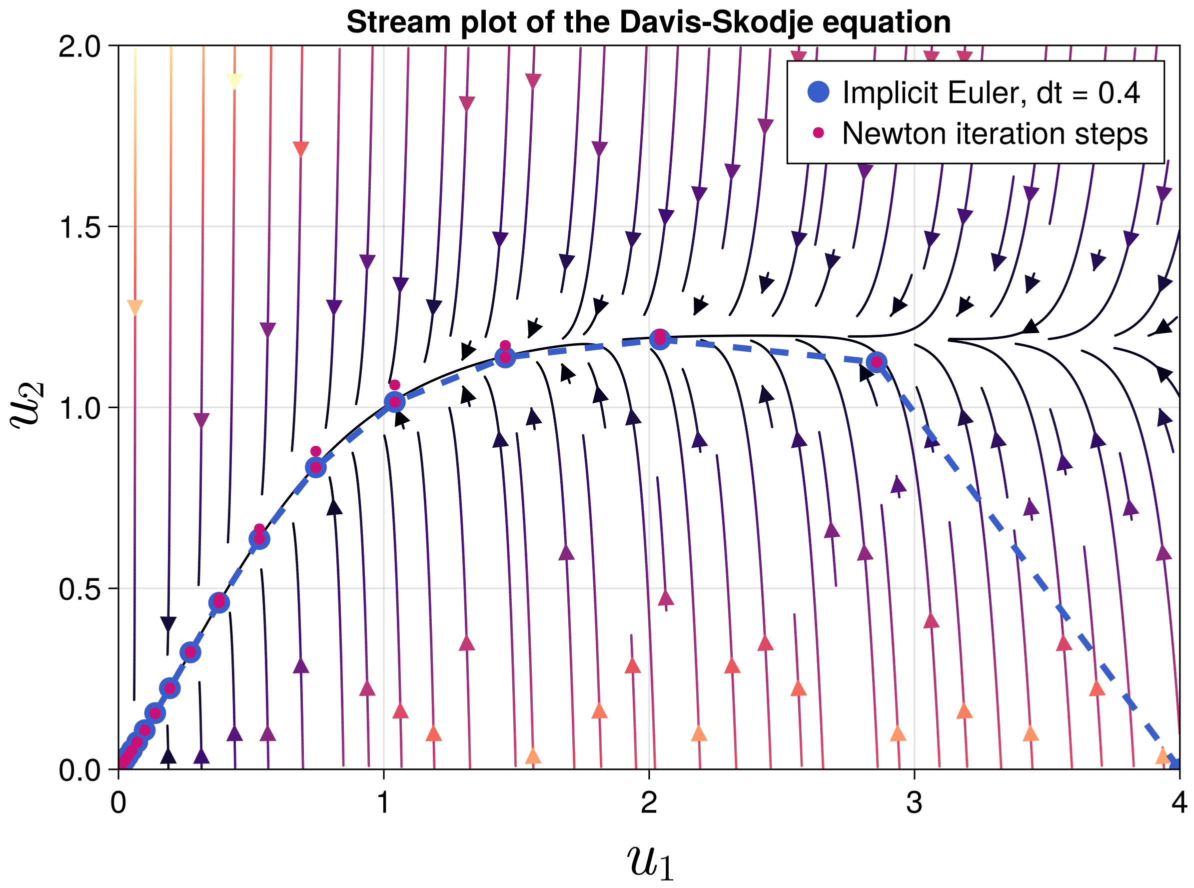


Figure 20: Stream plot of the Davis-Skodje equation with some Implicit Euler steps also drawn. The direction of the Implicit Euler steps is from right (starting at (4, 0)) to left.

3.5 Construction of higher-order methods

The two basic kinds of numerical methods for ODEs are

- one-step methods using one starting value at each step (in a step we want to get from $y^{(n)}$ to $y^{(n+1)}$)
- multistep methods using e.g. also $y^{(n-1)}, y^{(n-2)}$ to get from $y^{(n)}$ to $y^{(n+1)}$

In this subsection, we essentially only deal with one-step methods.

3.5.1 Meaning of going to higher order

Let us first understand, why we would want a higher order method. We still want to solve $\underline{f} = \underline{f}(\underline{y}, t)$ with $\underline{y}(t^{(0)}) = \underline{y}^{(0)}$ and our scheme approximates $\underline{y}(t^{(0)} + h)$ as $\underline{y}^{(1)}$. The scheme has order p if

$$||\underline{y}(t^{(0)} + h) - \underline{y}^{(1)}|| \leq K h^{p+1} \tag{56}$$

for sufficiently smooth problems, so if the Taylor series for the exact solution $\underline{y}(t^{(0)}+h)$ and for $\underline{y}^{(1)}$ coincide up to (and including) h^p.

Computational advantage: Our basic unit of cost are function evaluations of $\underline{f}(\underline{y},t)$, where explicit Euler takes one function evaluation per step and has $p=1$. Now imagine we can construct a second order scheme ($p=2$) with some constant number of function evaluations per step. While halving the step-size still doubles the integration cost over an interval for $p=2$ it quarters the error, so at some point the higher order scheme will be advantageous.

3.5.2 Approaches to constructing a higher order method

Higher order schemes can either be constructed using Taylor expansion so higher order derivatives (can be costly) or by the weighted combination of simple derivatives of multiple points and clever substeps.

3.5.3 Construction by Taylor expansion

The most obvious way to get a higher order truncation error, is to build a scheme based on higher order Taylor expansion.

We start by expanding $y(t)$ around some time t.

$$y(t+h)=y(t)+h\,\partial_t y\big|_t+\frac{h^2}{2}\,\partial_t^2 y\big|_t+\frac{h^3}{6}\,\partial_t^3 y\big|_t+\mathcal{O}(h^4) \tag{57}$$

The expansion up to the first order is just our explicit Euler scheme. As of our problem statement (1st order ODE)

$$\partial_t y=f(y,t),\quad y(t^{(0)})=y^{(0)},\quad t^{(n)}=t^{(0)}+hn,\quad \text{stepsize } h \tag{58}$$

(with the solutions defining 2D trajectories $(t,y(t))$ and we approximate $(t^{(n)},y^{(n)})$), we can express the higher order derivatives in the Taylor expansion as (chain rule)

$$\begin{gathered}\partial_t^k y\big|_t=\left(\frac{d}{dt}\right)^{k-1} f\Bigg|_{y(t),t}=:f^{(k-1)}(y,t)\\ f^{(k)}(y,t)=\partial_t f^{(k-1)}(y,t)+(\partial_t y)\partial_y f^{(k-1)}(y,t)=\partial_t f^{(k-1)}(y,t)+f\partial_y f^{(k-1)}(y,t)\end{gathered} \tag{59}$$

Problem: The higher order derivatives have to be calculated recursively e.g. with forward differentiation at the ground level which is complicated and slow.

Problem: In high-order Taylor expansion, the individual terms can be numerically problematic, e.g. in

$$\cos(x)|_{x=0} \approx 1 - \frac{x^2}{2} + \frac{x^4}{4!} + \dots \tag{60}$$

all polynomial terms diverge while the infinite sum is bound in $[-1, 1]$ as expected to the cosine.

3.5.4 Runge-Kutta (RK) Integration schemes I: General Idea

Consider instead of a 1st order ODE problem $\partial_t y = f(y, t)$ we had a quadrature problem $\partial_t y = f(t)$. Then a step would be

$$y(t+h) = y(t) + \int_t^{t+h} f(t')dt' \tag{61}$$

where we could approximate, e.g. using the trapezoidal rule

$$y^{(n+1)} = y^{(n)} + h\frac{f^{(n+1)} + f^{(n)}}{2}, \quad f(n) = f(t^{(0)} + hn) \tag{62}$$

We can't just apply this to the ODE $\partial_t y = f(y, t)$, because to calculate $f^{(n+1)}$ there we need $y^{(n+1)}$ which is what we are searching for. But what if we would approximate $y^{(n+1)}$ for $f^{(n+1)}$ with an Euler step? We would then have

$$\begin{aligned} k_1 &= f\left(y^{(n)}, t^{(n)}\right) \\ k_2 &= f\left(y^{(n)} + hk_1, t^{(n)} + h\right) \\ y^{(n+1)} &= y^{(n)} + \frac{h}{2}(k_1 + k_2) + \mathcal{O}\left(h^3\right) \end{aligned} \tag{63}$$

where the central advantage roughly is, that k_2 which includes the Euler approximation of $\mathcal{O}(h^2)$ is multiplied by h in the expression for $y^{(n+1)}$ so the error becomes less important. This already is the *RK2* scheme. The more general idea is illustrated in figure 21.

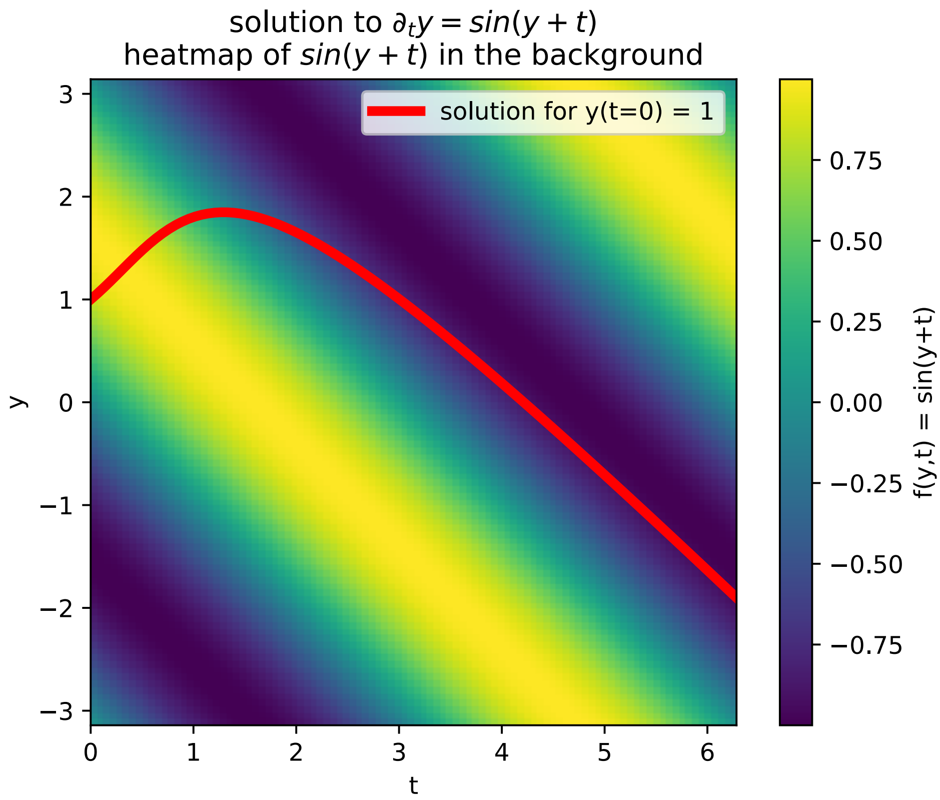


Figure 21: For a step in t the corresponding step in y equals the integration of $f(y,t)$ over the correct path. The correct path, however, is unknown. So we approximate points along the path to get a better approximation of the step as an approximation of the integral over $f(y,t)$.

3.5.5 Runge-Kutta (RK) Integration schemes II: Derivation of the general RK scheme

Using clever substeps, we want to construct an accurate and cost-efficient scheme, to solve the initial value problem $\partial_t \underline{y} = \underline{f}(\underline{y}, t)$ with $y(t^{(0)}) = y^{(0)}$. We make the ansatz

$$\begin{aligned}
\underline{y}(t+h) &= \underline{y}(t) + \left[\underline{y}(t+h) - \underline{y}(t)\right] \\
&= \underline{y}(t) + \int_t^{t+h} \partial_t \underline{y}\big|_{t'} \, dt' \\
&= \underline{y}(t) + h \int_0^1 \partial_t \underline{y}\big|_{t+\tau h} \, d\tau \\
&= \underline{y}(t) + h \int_0^1 \underline{f}(\underline{y}(t+\tau h), t+\tau h) d\tau
\end{aligned} \tag{64}$$

which will later allow us to extrapolate $\underline{y}$ according to the ODE. We approximate the integral by a quadrature rule of the form

$$\begin{aligned} \int_0^1 g(\tau)d\tau &\approx \sum_{i=1}^m \beta_i g(\gamma_i), \quad \text{RK weights } \beta_i, \quad \text{RK nodes } \gamma_i \\ \sum_{i=1}^m \beta_i = 1 \quad &\rightarrow \quad \text{correct integration of unity } g \equiv 1, \quad \text{scheme order } m \end{aligned} \tag{65}$$

Application to the ansatz (64) yields

$$\underline{y}(t+h) \approx \underline{y}(t) + h \sum_{i=1}^m \beta_i \underline{f}(\underline{y}(t+\gamma_i h), t+\gamma_i h) \tag{66}$$

Problem: We don't know $\underline{y}(t+\gamma_i h)$ yet.

Idea: Use an analogous quadrature rule to get approximations to $\underline{y}(t+\gamma_i h)$ with the γ_i as nodes again - quadrature in quadrature.

$$\begin{aligned} \underline{y}(t+\gamma_i h) = \underline{y}(t) + h \int_0^{\gamma_i} \partial_t \underline{y}\big|_{t+\tau h} d\tau &\approx \underline{y}(t) + h \sum_{l=1}^m \alpha_{i,l}\, \partial_t \underline{y}\big|_{t+\gamma_l h} \\ \sum_{i=1}^m = \gamma_i \quad &\rightarrow \quad \text{correct integration of unity } g \equiv 1, \quad i = 1, \dots, m \end{aligned} \tag{67}$$

We define

$$\underline{k}_l := \partial_t \underline{y}\big|_{t+\gamma_l h} = \underline{f}\left(\underline{y}(t+\gamma_l h), t+\gamma_l h\right) \tag{68}$$

But what have we won, we still need the $\underline{y}(t+\gamma_l h)$, right? By setting $\alpha_{i,l} = 0$ for $l \leq i$, we gain an explicit scheme where $\underline{y}(t+\gamma_l h)$ is constructed only based on previous substeps.

1. Starting with $\underline{y}^{(0)} = \underline{y}(t^{(0)})$ and $\underline{k}_1 = \underline{f}(\underline{y}^{(0)}, t^{(0)})$ we approximate $\underline{y}(t^{(0)} + \gamma_1 h)$ from which we calculate $\underline{k}_1 = \underline{f}(\underline{y}(t^{(0)}+\gamma_1 h), t^{(0)}+\gamma_1 h)$, then based on k_0, k_1 we approximate $\underline{y}(t^{(0)} + \gamma_2 h)$ and thus k_3 and so on.

2. Based on $\underline{k}_1, \dots, \underline{k}_m$, we approximate $\underline{y}^{(1)} = \underline{y}^{(0)} + h \sum_{i=1}^m \beta_i \underline{k}_i$

3. ...

3.5.6 Runge-Kutta (RK) Integration schemes III: General m-substep RK method

In general, we have obtained

$$
\begin{aligned}
\underline{y}^{(n+1)} &= \underline{y}^{(n)} + h\sum_{i=1}^{m} \beta_i \underline{k}_i \\
\underline{k}_i &= f\left(\left(\underline{y}^{(n)} + h\sum_{l=1}^{m-1} \alpha_{i,l}\underline{k}_l\right), t_n + \gamma_i h\right), \quad i = 1, \dots, m \\
\sum_{l=1}^{m} \alpha_{i,l} &= \gamma_i, \quad \alpha_{i,l} = 0 \text{ for } l \geq i \rightarrow \text{ explicit method}
\end{aligned}
\tag{69}
$$

where for $\alpha_{i,l} = 0$ for $l \geq i$, $\underline{k}_i$ only depends on $\underline{k}_l, l < i$.

3.5.6.1 Butcher-Tableau for visualizing the RK coefficients

The general Butcher-Tableau is given in figure 22, for an explicit scheme, the α's form a lower left triangular matrix ($\alpha_{i,l} = 0$ for $l \geq i$, $\underline{k}_i$).

Examples for butcher tableaus of explicit RK methods are given in table 3.

<table>
<tr><th>Explicit Euler
(1st order)</th><th>Implicit Euler
(1st order)</th><th>RK2 (2nd order)</th><th>RK4 (4th order)</th></tr>
<tr>
<td>$\begin{array}{c|c} 0 & \\ \hline & 1 \end{array}$</td>
<td>$\begin{array}{c|c} 0 & 1 \\ \hline & 1 \end{array}$</td>
<td>$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$</td>
<td>$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$</td>
</tr>
</table>

Table 3: Butcher-Tableau for explicit RK methods.

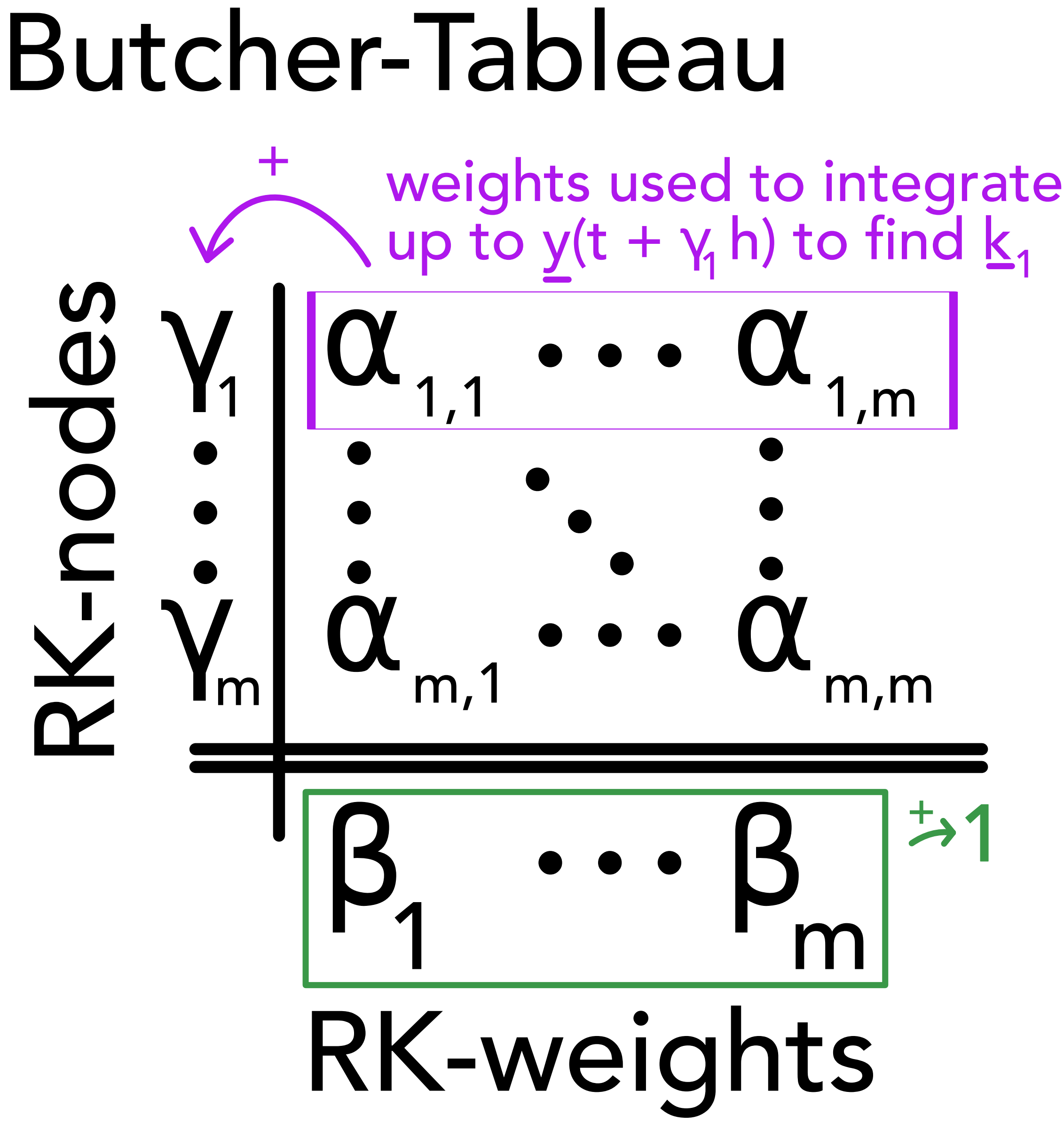


Figure 22: Butcher-Tableau for the general m-substep RK method.

3.5.7 Runge-Kutta (RK) Integration schemes IV: Taylor expansion to identify RK parameters for 2nd order schemes

3.5.7.1 Comparison of coefficients

We want to find appropriate α's and β's such that the RK scheme follows the **Taylor expansion**

$$\begin{aligned} y(t^{(n+1)}) &= y^{(n+1)} \\ &= y^{(n)} + h\partial_t y + \frac{h^2}{2}\partial_t^2 y + \mathcal{O}(h^3) \\ &= y^{(n)} + hf + \frac{h^2}{2}\frac{d}{dt}f + \mathcal{O}(h^3) \\ &= y^{(n)} + hf + \frac{h^2}{2}\left((\partial_y f)\partial_t y + \partial_t f\right) + \mathcal{O}(h^3) \\ &\boxed{= y^{(n)} + hf + \frac{h^2}{2}\left(f\partial_y f + \partial_t f\right) + \mathcal{O}(h^3)} \end{aligned} \tag{70}$$

where if not specified differently, the evaluation is at (y_n, t_n), so that we know that the error per step is $\mathcal{O}(h^3)$.

Now let us bring the explicit RK ansatz for $m = 2$ ($\rightarrow$ only $\alpha_{2,1} \neq 0$ so $\gamma_1 = 0$ and $\gamma_2 = \alpha_{2,1}$) into the form of the Taylor expansion. We start with

$$\begin{aligned} y^{(n+1)} &= y^{(n)} + h(\beta_1 k_1 + \beta_2 k_2) \\ &= y^{(n)} + h\left(\beta_1 f + \beta_2 f\left((y^{(n)} + h\alpha_{2,1} f), t^{(n)} + \gamma_2 h\right)\right) \end{aligned} \tag{71}$$

and first order expand k_2 to

$$f\left((y^{(n)} + h\alpha_{2,1} f), t^{(n)} + \gamma_2 h\right) = f + h\alpha_{2,1} f\partial_y f + \gamma_2 h\partial_t f \tag{72}$$

Using $\gamma_2 = \alpha_{2,1}$ yields a form that allows comparison of coefficients

$$\begin{aligned} y^{(n+1)} &= y^{(n)} + h\cdot\left(\beta_1 f + \beta_2\left(f + h\alpha_{2,1} f\partial_y f + \alpha_{2,1} h\partial_t f\right)\right) \\ &\boxed{= y^{(n)} + hf\cdot(\beta_1 + \beta_2) + h^2\beta_2\alpha_{2,1}\left(f\partial_y f + \partial_t f\right)} \end{aligned} \tag{73}$$

Comparing the boxed equations yields two equations for three variables $(\beta_1, \beta_2, \alpha_{2,1})$

$$\beta_1 + \beta_2 = 1, \quad \beta_2\alpha_{2,1} = \frac{1}{2} \quad \rightarrow \quad \text{choose } \beta_2 = q \tag{74}$$

so

$$\beta_1 = 1 - q, \quad \beta_2 = q, \quad \alpha_{2,1} = \frac{1}{2q} \tag{75}$$

3.5.7.2 Resulting integration formula with free parameter q

We get the integration formula

$$t^{(n+1)} = t^{(n)} + h, \quad y^{(n+1)} = y^{(n)} + h\left[(1-q)f + qf\left((y^{(n)} + \frac{h}{2q}f), t^{(n)} + \frac{h}{2q}\right)\right] \tag{76}$$

3.5.7.3 Different integration schemes based on the choice of q

Depending on q we can yield different schemes, with examples shown in table 4.

Only based on two function evaluations of f, the midpoint rule and RK2 are already second order accurate.

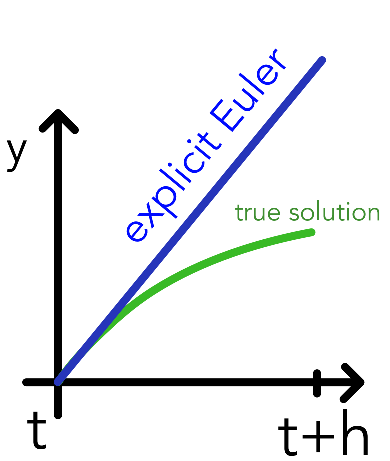
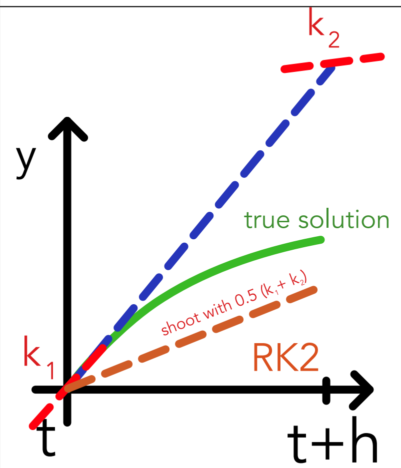
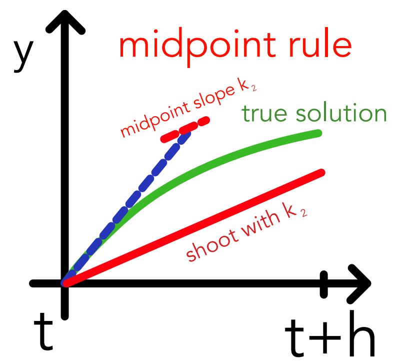
<table>
<tr><th>$q = 0$</th><th>$q = \frac{1}{2}$</th><th>$q = 1$</th></tr>
<tr><td>Forward Euler</td><td>2nd order Runge-Kutta (RK2) (aka Heun-method)</td><td>Midpoint-Rule</td></tr>
<tr><td>$$k_1 = f(y^{(n),t^{(n)}})$$ $$y^{(n+1)} = y^{(n)} + hk_1 + \mathcal{O}(h^2)$$</td><td>$$k_1 = f(y^{(n),t^{(n)}})$$ $$k_2 = f(y^{(n)} + hk_1, t^{(n)} + h)$$ $$y^{(n+1)} = y^{(n)} + \frac{h}{2}(k_1 + k_2) + \mathcal{O}(h^3)$$</td><td>$$k_1 = f(y^{(n),t^{(n)}})$$ $$k_2 = f(y^{(n)} + \frac{h}{2}k_1, t^{(n)} + \frac{h}{2})$$ $$y^{(n+1)} = y^{(n)} + hk_2 + \mathcal{O}(h^3)$$</td></tr>
<tr><td>Shoot along a tangent from the starting point.</td><td>Using k_1 Euler-approximate $y(t+h)$
Find k_2 using this approximation
Shoot with the mean of k_1 and k_2</td><td>Approximate y at the mid-point of the interval and use the slope there for shooting across the whole interval.</td></tr>
<tr><td>
</td><td>
</td><td>
</td></tr>
</table>

Table 4: Different schemes based on the choice of q.

3.5.8 Runge-Kutta (RK) Integration schemes V: Classical 4th order RK scheme (RK-4)

$$\begin{gathered}
\underline{k}_1 = \underline{f}(\underline{y}^{(n)}, t^{(n)}), \quad \underline{k}_2 = \underline{f}(\underline{y}^{(n)} + \frac{h}{2}\underline{k}_1, t^{(n)} + \frac{h}{2}), \\
\underline{k}_3 = \underline{f}(\underline{y}^{(n)} + \frac{h}{2}\underline{k}_2, t^{(n)} + \frac{h}{2}), \quad \underline{k}_4 = \underline{f}(\underline{y}^{(n)} + h\underline{k}_3, t^{(n)} + h) \\
\underline{y}^{(n+1)} = \underline{y}^{(n)} + \frac{h}{6}\left(\underline{k}_1 + 2\underline{k}_2 + 2\underline{k}_3 + \underline{k}_4\right) + \mathcal{O}(h^5)
\end{gathered} \tag{77}$$

Note: Here we need 4 function evaluations of f per step. Depending on the situation, lower order schemes with smaller stepsize might be more efficient. Choosing an appropriate step-size might is very important.

An example application of RK4 with the corresponding evaluation points of k_2, k_3, k_4 is shown in figure 23.

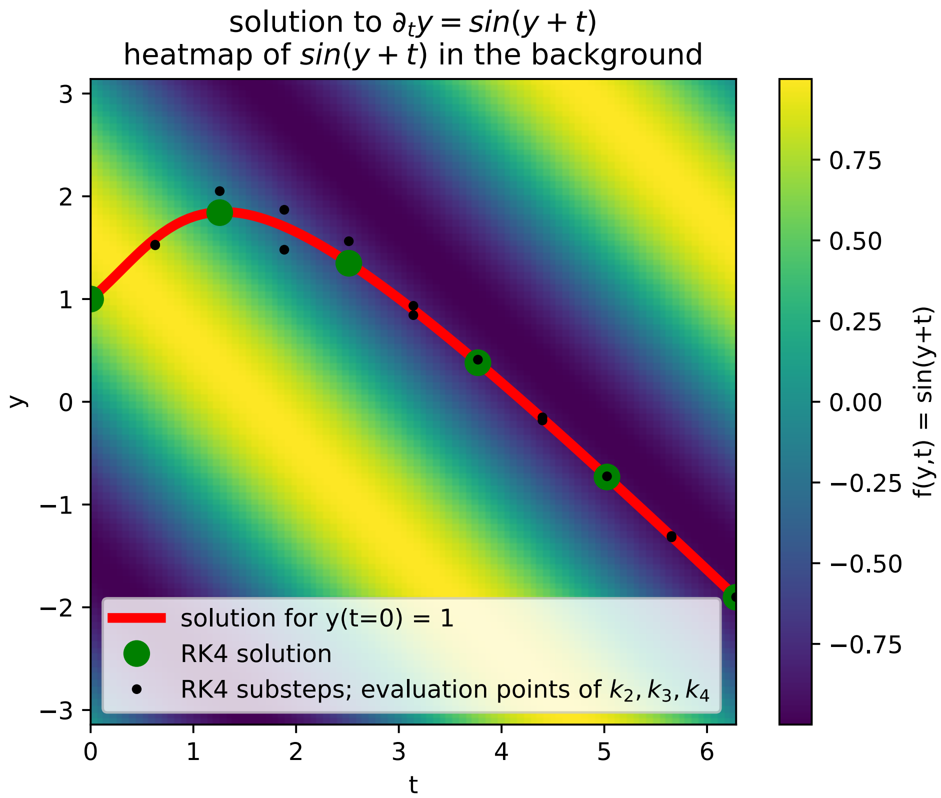


Figure 23: RK4 applied to the problem $\partial_t y = f(y,t)$ with $f(y,t) = \sin(y+t)$ and $y(0) = 1$. The evaluation points of k_2, k_3, k_4 are marked in black.

3.6 Adaptive Step Sizes

Problem: While in one region of a problem a relatively large step-size might suffice in others we might need a very small one. Using the large step size everywhere does not work but taking the small one everywhere is a waste of compute.

Idea: Take steps of adaptive size, striking a balance between accuracy, stability and efficiency. The adaptation is based on the estimation of a local integration error and a user-specified wanted upper bound on it. For an example see figure 24.

When we simulate e.g. hydrogen in space we might even want to use different step-sizes in different local regions of the problem (smaller timesteps in denser regions).

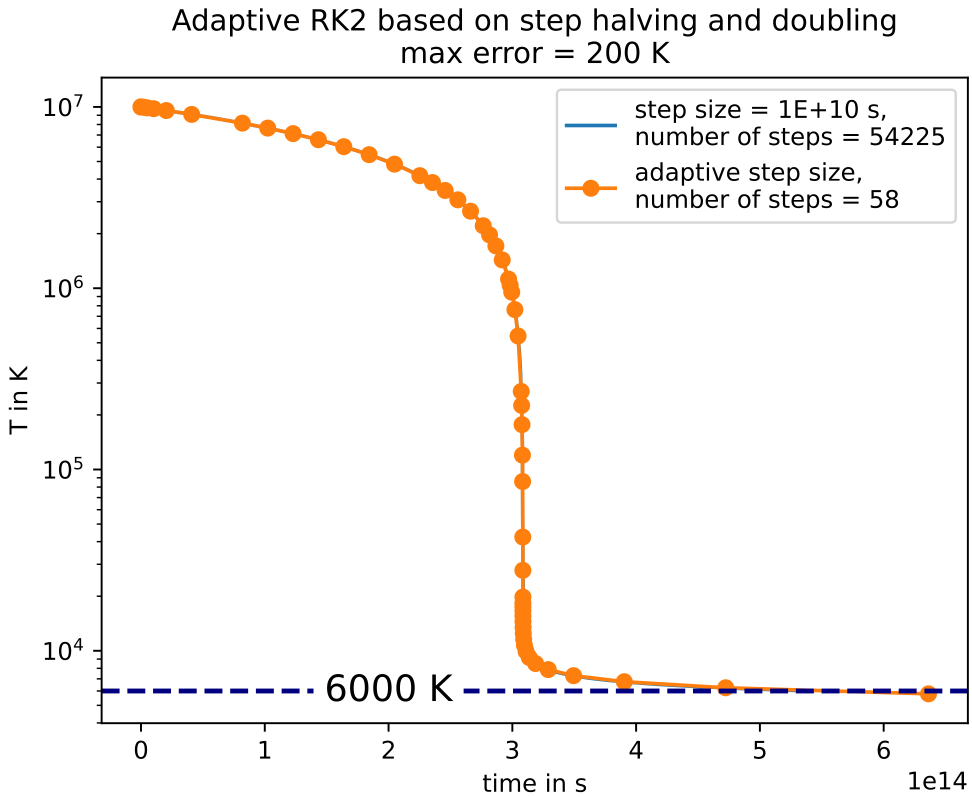


Figure 24: Example of adaptive step sizes.

3.6.1 Step halving and doubling method

We can estimate the local integration error by

- performing one step of size h to obtain y_a
- performing two steps of size $\frac{h}{2}$ to obtain y_b

from the same starting point and then comparing the results.

With this, the **halving and doubling scheme** given the user-specified local upper error bound ϵ_0 is

1. Calculate y_a, y_b and $\epsilon = |y_a - y_b|$
2. If $\epsilon > \epsilon_0$ discard the step and try again with $h' = \frac{h}{2}$
3. If $\epsilon \ll \epsilon_0$, keep y_b and use $h' = 2h$ for the next step (doubling)
4. Else if $\epsilon < \epsilon_0$, keep y_b and retain h for the next step

Advantage of the halving and doubling scheme in spatio-temporal simulations: Consider some hydrogen simulation. In a halving-doubling scheme we can use different step sizes in different spatial regions and still have results in sync - for every point on the coarse time-grid we will also have a result from the finer grids in time, see figure 25

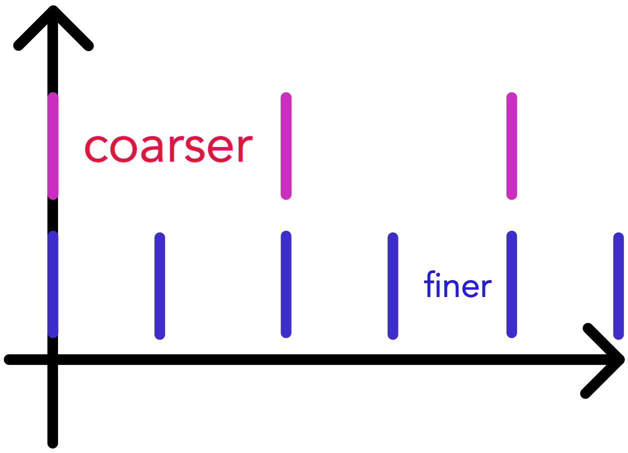


Figure 25: Different step sizes in time but still results in sync in the halving-doubling scheme.

Note: We said we want to double h for the next step for $\epsilon \ll \epsilon_0$. A more concise criterion is even if we double we would expect an error lower than the user-specified upper bound. For this we first make a remark on the local accuracy.

3.6.2 Note on the Local accuracy

As a p-th order scheme is locally accurate to the $p+1$-th order, we can write the local errors as

$$\begin{aligned} y_a - y(t^{(0)} + h) &= \alpha h^{p+1} + \mathcal{O}(h^{p+2}) \\ y_b - y(t^{(0)} + h) &= 2\alpha \left(\frac{h}{2}\right)^{p+1} + \mathcal{O}(h^{p+2}) \end{aligned} \tag{78}$$

yielding an error estimate of

$$\epsilon = |y_a - y_b| = \alpha h^{p+1}(1 - 2^{-p}) \tag{79}$$

3.6.3 When does doubling make sense?

For a doubled time-step, we expect the error ($h \to 2h$ in (79)) to be

$$\epsilon' = \alpha (2h)^{p+1}(1 - 2^{-p}) = 2^{p+1}\epsilon \tag{80}$$

which we still want to be smaller than ϵ_0 so we double if

$$\epsilon' = 2^{p+1}\epsilon < \epsilon_0 \tag{81}$$

so when the expected error after doubling is below our error bound.

3.6.4 Adaptively choosing ϵ_0

Problem: The smaller the time-step the more steps we need to cover a certain timespan, the more error can accumulate.

Idea: Set ϵ_0 adaptively, taking into account how many timesteps one would need to cover the total integration time with the current h, so

$$\epsilon_0 = \frac{h}{T}\epsilon_0^{\text{global}}, \quad \text{total integration time } T, \quad \text{prescribed global error bound } \epsilon_0^{\text{global}} \tag{82}$$

3.6.5 Continuous time step adjustment

While the halving-doubling is nicely suited for being able to use different step-sized in different spatial regions of a simulation, often we want a more flexible continuous adjustment.

Idea: For the next step use a step-size h^{new} such that we would assume this step-size to just hit the error bound in our current step (with some safety factor.)

The next timestep is scaled according to the current error, such that for h^{desired} we have $\epsilon' = \epsilon_0$ so

$$\begin{aligned} \epsilon_0 = \epsilon' = \alpha \cdot \left(h^{\text{desired}}\right)^{p+1} \cdot (1 - 2^{-p}) = \left(\frac{h^{\text{desired}}}{h}\right)^{p+1} \epsilon \\ \to h^{\text{desired}} = h\left(\frac{\epsilon_0}{\epsilon}\right)^{\frac{1}{p+1}}, \quad \text{error } \epsilon \text{ if step with size } h \text{ is taken} \end{aligned} \tag{83}$$

Note that we have used the error formula for the halving-doubling scheme but more generally assuming $\mathcal{O}(\epsilon) = h^{p+1}$ we get the same result for h^{desired} from

$$\frac{\epsilon_0}{\epsilon} = \left(\frac{h^{\text{desired}}}{h}\right)^{p+1} \tag{84}$$

3.6.5.1 Continuous adaptive time step control scheme

We get to the scheme

1. Advance the system by a step h and extimate the error of the step, we assume a p-th

order scheme with $\mathcal{O}(\epsilon) = h^{p+1}$

2. Calculate the new step size as

$$h^{\text{new}} = \beta h \left(\frac{\epsilon_0}{\epsilon}\right)^{\frac{1}{p+1}} \tag{85}$$

with some safety factor $\beta \sim 0.9$

3. If $\epsilon < \epsilon_0$ accept the step, otherwise discard it and repeat with the new step-size

But is there a more efficient way to estimate ϵ than the halving-doubling scheme?

3.6.5.2 Embedded Runge-Kutta schemes for cheaper error estimates

In embedded Runge-Kutta schemes like Runge-Kutta-Fehlberg (e.g. RKF45), based on the same function evaluations and respectively $\underline{k}_i$ schemes of different order are constructed and from the difference of their results, the error is estimated.

For instance RK45 is illustrated in figure 26.

RK nodes

RKF45

$$k_1 = f(y^{(n)}, t^{(n)})$$

$$k_2 = f(y^{(n)} + \frac{1}{4} h k_1, t^{(n)} + \frac{1}{4} h)$$

$$k_3 = f(y^{(n)} + \frac{3}{32} h k_1 + \frac{9}{32} h k_2, t^{(n)} + \frac{3}{8} h)$$

$$
\left[\begin{array}{c|cccccc}
0 & & & & & & \\
\frac{1}{4} & \frac{1}{4} & & & & & \\
\frac{3}{8} & \frac{3}{32} & \frac{9}{32} & & & & \\
\frac{12}{13} & \frac{1932}{2197} & -\frac{7200}{2197} & \frac{7296}{2197} & & & \\
1 & \frac{439}{216} & -8 & \frac{3680}{513} & -\frac{845}{4104} & & \\
\frac{1}{2} & -\frac{8}{27} & 2 & -\frac{3544}{2565} & \frac{1859}{4104} & -\frac{11}{40} & \\
\hline
 & \frac{25}{216} & 0 & \frac{1408}{2565} & \frac{2197}{4104} & -\frac{1}{5} & 0 \\
\hline
 & \frac{16}{135} & 0 & \frac{6656}{12825} & \frac{28561}{56430} & -\frac{9}{50} & \frac{2}{55}
\end{array}\right]
\begin{array}{l}
\\ \\ \\ \\ \\ \\
\rightarrow y_A^{(n+1)} \\
\rightarrow y_B^{(n+1)}
\end{array}
$$

RK weights

$$\varepsilon = |y_A^{(n+1)} - y_B^{(n+1)}|$$

Figure 26: Embedded Runge-Kutta scheme RK45.

3.7 The problem of conserved quantities | Symplectic Integrators

3.7.1 Hamiltonian Systems and Symplecticity

Evolutions of classical physical systems can very generally be stated in terms of equations of motion derived from a real-valued, smooth Hamiltonian $H(\underline{p}, \underline{q})$

$$\begin{aligned}\partial_t \underline{p} &= -\underline{\nabla}_{\underline{q}} H(\underline{p}, \underline{q}) \\ \partial_t \underline{q} &= \underline{\nabla}_{\underline{p}} H(\underline{p}, \underline{q})\end{aligned} \quad \text{or} \quad \partial_t \underline{y} = \underline{\underline{J}}^{-1} \underline{\nabla} H(\underline{y}), \underline{y} = \begin{pmatrix} \underline{p} \\ \underline{q} \end{pmatrix}, \quad \underline{\underline{J}} = \begin{pmatrix} 0 & \underline{\underline{1}} \\ -\underline{\underline{1}} & 0 \end{pmatrix} \in \mathbb{R}^{2d} \tag{86}$$

where $\underline{p} \in \mathbb{R}^d$ is the generalized momentum and $\underline{q} \in \mathbb{R}^d$ are the generalized coordinates. The Hamiltonian can be followed as the legendre transform of the Lagrangian.

Hamiltonian systems

- conserve energy, i. e. $H(\underline{p}, \underline{q})$ is conserved along a trajectory if the Hamiltonian does not explicitly depend on time
- show *symplecticity*, which is most intuitively understood as area conservation in phase space Hairer, Wanner, and Lubich, 2006, phase space volumina spanned by trajectories remain constant (see the following examples) / the phase space ditribution function is constant along trajectories

3.7.1.1 Poisson brackets and constants of motion (first integrals)

The Poisson brackets are

$$\{f, g\} = \{f, g\}_{qp} = \sum_{k=1}^{d} \frac{\partial f}{\partial q_k} \frac{\partial g}{\partial p_k} - \frac{\partial f}{\partial p_k} \frac{\partial g}{\partial q_k} \tag{87}$$

and we can find the **total derivation of a variable** $F(p, q, t)$ **along physical trajectories** as

$$\frac{dF(p(t), q(t), t)}{dt} = \{F, H\} + \partial_t F \tag{88}$$

so F is a constant of motion if $\{F, H\} = 0$ and $\partial_t F = 0$

3.7.1.2 Canonical transformations

Consider a coordinate transform in phase space

$$(p, q, t) \quad \rightarrow \quad (P, Q, t), \quad Q_i = Q_i(p, q, t), \quad P_i = P_i(p, q, t), \quad i = 1, \ldots, d \tag{89}$$

and the corresponding transformation of the Hamiltonian

$$H(p, q, t) \quad \rightarrow \quad \tilde{H}(P, Q, t) \tag{90}$$

Then if the coordinate transformation is a **canonical transformations** it necessarily leaves the Hamiltonian equations of movement invariant.

$$\begin{aligned} \partial_t \underline{P} &= -\underline{\nabla}_{\underline{Q}} \tilde{H}(\underline{P}, \underline{Q}) \\ \partial_t \underline{Q} &= \underline{\nabla}_{\underline{P}} \tilde{H}(\underline{P}, \underline{Q}) \end{aligned} \tag{91}$$

Closely connected are the Hamilton-Jacobi equations.

3.7.1.3 Definition of symplectic transformations

Linear maps: A linear map in 2D $F : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ is called symplectic if $\forall \underline{\xi}, \underline{\eta} \in \mathbb{R}^{2d}$: $\omega(\underline{\underline{F}}\,\underline{\xi}, \underline{\underline{F}}\,\underline{\eta}) = \omega(\underline{\xi}, \underline{\eta})$, where ω gives the area of the parallelogram spanned by both vectors.

Differentiable maps: $g : U \to \mathbb{R}^{2d}$ is symplectic if its Jacobian matrix $\underline{\underline{J}}_g$ is everywhere symplectic, so $\omega(\underline{\underline{J}}_g \underline{\xi}, \underline{\underline{J}}_g \underline{\eta}) = \omega(\underline{\xi}, \underline{\eta})$.

Connection to canonical transformations: In Hamiltonian systems, symplectic transformations are canonical transformations.

Compositions: Compositions of symplectic transformations are symplectic again.

Poincare's recurrence theorem: The time-evolution generated by a Hamiltonian in phase space is a symplectic transformation.

Idea: If the Hamiltonian evolution is symplectic, maybe its advantageous if our integrator is symplectic too.

3.7.2 Runge-Kutta methods do not conserve energy and are not symplectic

Let us now apply RK2 to the two-body-problem reduced to a dimensionless one-body-problem as described by

$$\frac{d\underline{s}}{d\tau} = \underline{w}, \quad \frac{d\underline{w}}{d\tau} = -\frac{\underline{s}}{s^3}$$

where $\underline{s}$ is the position and $\underline{w}$ the velocity. The numerical solution obtained using RK2 can be seen in figure 27. One can see that neither the energy nor the angular momentum nor the Runge-Lenz vector are conserved. Every step in the scheme is erroneous, and those errors add up leading to our quantities of interest not being conserved.

Let us look at another physical problem, the pendulum, as it lends itself well to phase space visualization. We see in figure 28 that the phase space area, the area spanned by the test points as they propagate in time, is not preserved as it should be for Hamiltonian flow.

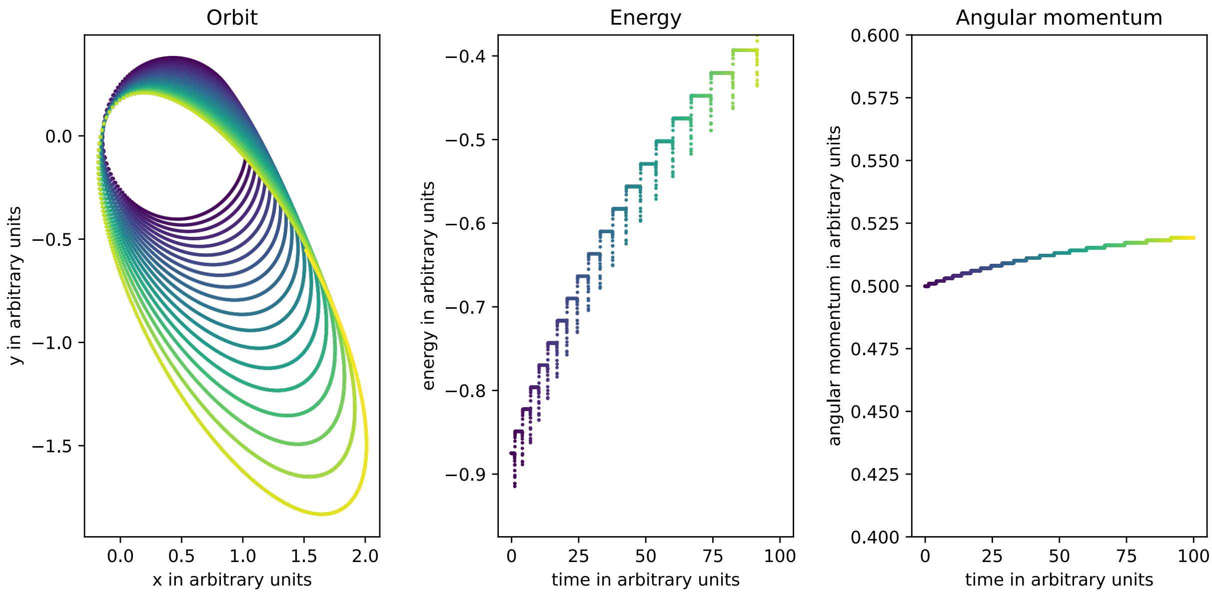


Figure 27: Numerical solution of the two-body-problem using RK2. The left panel shows the orbit, the central one the energy and the right one the angular momentum. Time is also encoded in the form of color, going from a dark blue to yellow.

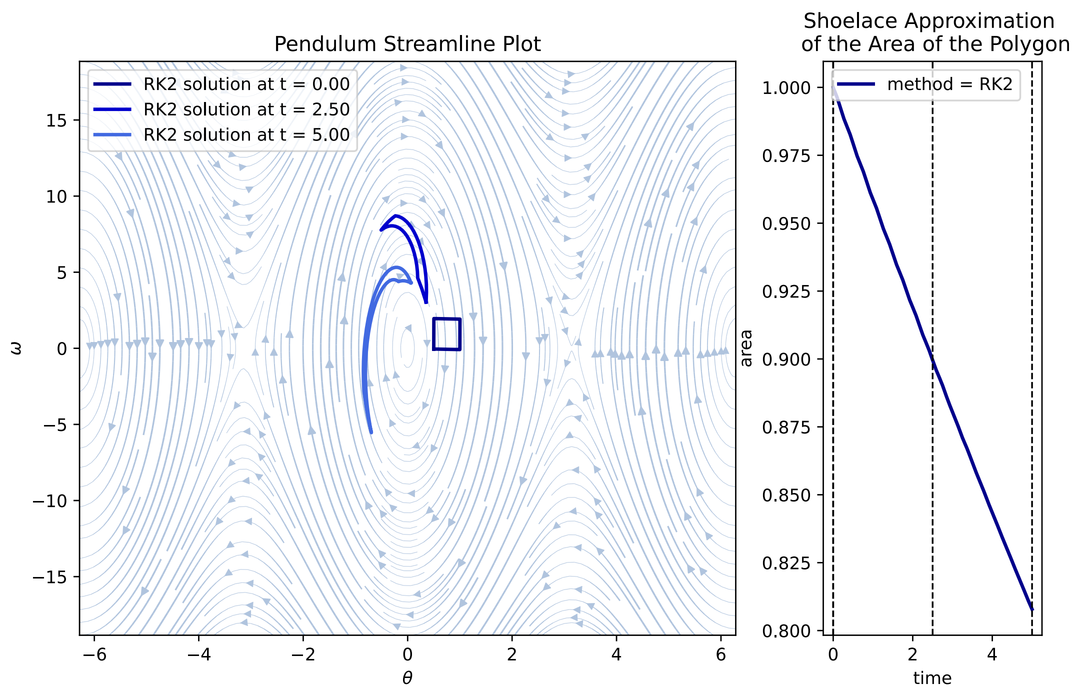


Figure 28: The left panel shows solutions to the pendulum problem at different points in time for different initial values as obtained by the RK2 scheme. The initial values are chosen so that they initially span a square in phase space. The right panel shows the phase space area spanned by the solutions as a function of time.

We need to switch to a whole other class of integrators.

3.7.3 Symplectic integrators to the help

This leads us to geometric integrators which are numerical methods preserving geometric properties of the exact flow of ODEs (Hairer, Wanner, and Lubich, 2006). More specifically for Hamiltonian systems we use **Symplectic Integrators** which produce a flow in phase space that is symplectic just as the flow of the exact solution (Hairer, Wanner, and Lubich, 2006, chapter VI).

Symplectic integrators

- are *structure preserving*
- nearly conserve properties of Hamiltonian systems, e.g. *first integrals* (variables $F(p, q)$ constant along the motion as dictated by the Hamiltonian)
- conserve phase-space (as prescribed by the Liouville theorem)
- more generally conserve all the Poincare invariants

It turns out that preserving symplecticity and energy at the same time is very difficult (Hairer, 2006). However, symplectic integrators still have good energy conservation properties without much long-term drifts. The general idea behind the connection between symplecticity and energy conservation is that geometric properties of the integrator (e.g. phase space conservation) translate into structure preservation on the level of modified equations (Hairer, Wanner, and Lubich, 2006, preface and chapters X through XII).

Problem: Symplectic integrators do not come without caveats:

- Using adaptive time-steps and keeping symplecticity is a problem.
- Non-conservative forces (those which cannot be described by a potential) like radiation forces (depending on velocity not only position) might be important, e.g. for debris in space and particles in planetary rings but then the central requirement of a Hamiltonian system breaks down
- The propagation of floating-point errors might be non-optimal

So the rebound N-body simulation package actually uses IAS15 (implicit integrator with adaptive time-stepping, 15th order) as the default (Rein and Spiegel, 2014).

Note: The default integrator in the N-body-simulator rebound is IAS15, a non-symplectic integrator

3.7.4 Verlet Scheme

Consider we want to solve the Newtonian equation of movement

$$\partial_t^2 \underline{s} = \underline{a}((\underline{s})) \tag{92}$$

- a classical physical problem with a particle at position $\underline{s}$ with velocity $\underline{v}$ and an acceleration $\underline{a}$ that depends only on the position.

Note: In the following we will introduce the Störmer-Verlet, velocity Verlet and Leapfrog scheme which differ in their formulations but are essentially the same. In molecular dynamics it is mostly called Verlet method, in the context of partial differential equations of wave propagation leapfrog method (Hairer, Lubich, et al., 2003).

We can derive a two-step scheme based on a Taylor expansion forward by Δt and backward by $-\Delta t$

$$\begin{aligned} \underline{s}(t+\Delta t) &= \underline{s}(t) + \underline{v}(t)\Delta t + \frac{1}{2}\underline{a}\Delta t^2 + \frac{1}{6}\underline{b}(t)\Delta t^3 + \mathcal{O}(\Delta t^4) \\ \underline{s}(t-\Delta t) &= \underline{s}(t) - \underline{v}(t)\Delta t + \frac{1}{2}\underline{a}\Delta t^2 - \frac{1}{6}\underline{b}(t)\Delta t^3 + \mathcal{O}(\Delta t^4) \end{aligned} \tag{93}$$

with

$$\begin{gathered} \text{position } \underline{s}, \quad \text{velocity } \underline{v}, \quad \text{acceleration} \underline{a} = -\frac{1}{m}\nabla V(\underline{s}) \\ \text{some potential } V, \quad \text{jerk } \underline{b} = \frac{d\underline{a}}{dt} \end{gathered} \tag{94}$$

Adding both equations yields a scheme 4th order accurate in time

$$\underline{s}(t+\Delta t) = 2\underline{s}(t) - \underline{s}(t-\Delta t) + \underline{a}(t)\Delta t^2 + \mathcal{O}(\Delta t^4) \tag{95}$$

where terms with odd powers of Δt were eliminated. This is a two-step scheme as for $\underline{s}(t+\Delta t)$, $\underline{s}(t)$ and $\underline{s}(t-\Delta t)$ are used (sometimes this form is called Störmer-Verlet).

Problem: What if we are also interested in calculating the velocity $\underline{v}$? What if $\underline{a}$ also depends on $\underline{v}$ as in the Lorentz and we need $\underline{v}$ to calculate $\underline{a}$?

From subtracting the equations 93, we can find

$$\underline{v}(t) = \frac{\underline{s}(t+\Delta t) - \underline{s}(t-\Delta t)}{2\Delta t} - \frac{1}{6}\underline{b}(t)\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{96}$$

but this is problematic as

- only accurate to the third order
- an implicit equation as we do not know the future $\underline{s}(t+\Delta t)$ at the current timepoint
- we would need to know the jerk at the current timestep which could itself depend on $\underline{v}(t)$, we could ignore the jerk, but then the accuracy is only $\mathcal{O}(\Delta t^2)$

3.7.4.1 Velocity Verlet algorithm

Idea: Starting again from the Taylor expansion, we omit the jerk and extrapolate the velocity $\underline{v}(t+\Delta t)$ using the average of the accelerations at t and $t+\Delta t$

$$\begin{aligned} \underline{s}(t+\Delta t) &= \underline{s}(t) + \underline{v}(t)\Delta t + \mathcal{O}(\Delta t)^3 \\ \underline{v}(t+\Delta t) &= \underline{v}(t) + \frac{\underline{a}(t) + \underline{a}(t+\Delta t)}{2} + \mathcal{O}(\Delta t)^2 \end{aligned} \tag{97}$$

Note: Again, we use $\underline{a}(t+\Delta t)$ to get $\underline{v}(t+\Delta t)$ which is not possible if $\underline{a}$ depends on v. If we would take only $\underline{a}(t)$ instead of the average, we would just have explicit Euler.

In steps we can write

1. calculate the new position: $\underline{s}(t+\Delta t) = \underline{(s)}(t) + \underline{v}(t)\Delta t + \frac{1}{2}\underline{a}(t)\Delta t^2$
2. update the acceleration: $\underline{a}(t+\Delta t) = -\frac{1}{m} \left.\nabla V\right|_{\underline{s}(t+\Delta t)}$
3. calculate the new velocity: $\underline{v}(t+\Delta t) = \underline{v}(t) + \frac{\underline{a}(t)+\underline{a}(t+\Delta t)}{2}\Delta t$
4. update the time: $t \to t + \Delta t$

Introducing half timesteps, we can write the Verlet scheme as a combination of 1st order steps

1. $\underline{v}\left(t+\frac{1}{2}\Delta t\right) = \underline{v}(t) + \frac{1}{2}\underline{a}(t)\Delta t$
2. $\underline{s}(t+\Delta t) = \underline{s}(t) + \underline{v}\left(t+\frac{1}{2}\Delta t\right)\Delta t$
   (note that plugging (1) into (2) yields the previous step (1))
3. $\underline{a}(t+\Delta t) = -\frac{1}{m} \left.\nabla V\right|_{\underline{s}(t+\Delta t)}$
4. $\underline{v}\left(t+\Delta t\right) = \underline{v}\left(t+\frac{1}{2}\Delta t\right) + \frac{1}{2}\underline{a}(t+\Delta t)\Delta t$
   (note that plugging (1) into (4) yields the previous step (3))
5. $t \to t + \Delta t$

This is illustrated in figure 29. Here the last step can be called *implicit* as it depends on $\underline{a}(t + \Delta t)$, so a result at the same time as it is supposed to deliver a result at - we have a semi-implicit Euler scheme.

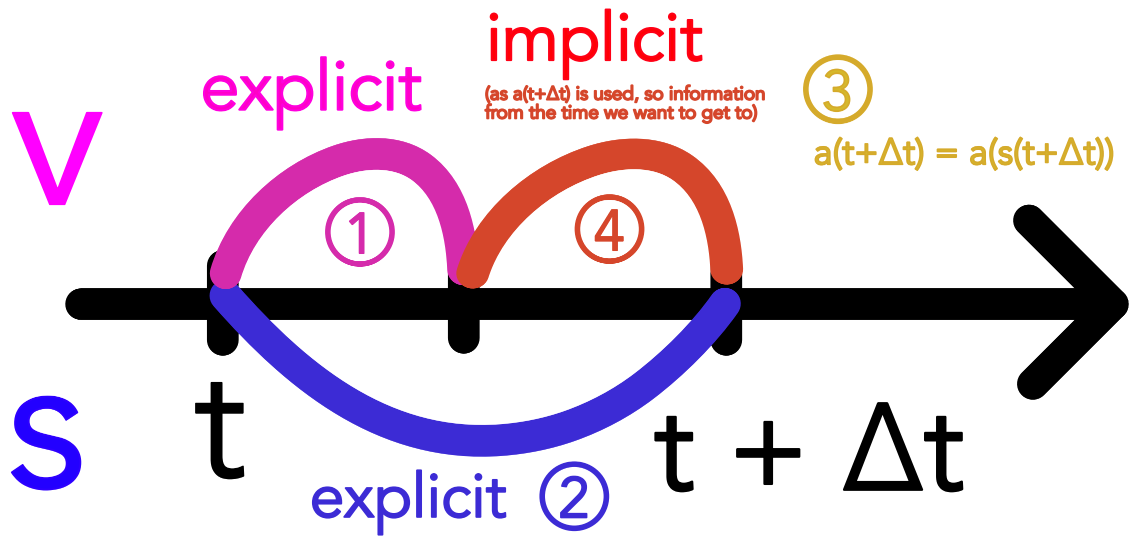


Figure 29: Illustration of the velocity Verlet scheme.

3.7.5 The Leapfrog Method

We will now introduce the Leapfrog scheme and at its hand what it means for a method to be symplectic.

The leapfrog scheme can then be written as

$$\begin{aligned} \underline{s}(t + \frac{1}{2}\Delta t) &= \underline{s}(t - \frac{1}{2}\Delta t) + \underline{v}(t)\Delta t + \mathcal{O}(\Delta t^3) \\ \underline{v}(t + \Delta t) &= \underline{v}(t) + \underline{a}(t + \frac{1}{2}\Delta t)\Delta t + \mathcal{O}(\Delta t^3) \end{aligned} \tag{98}$$

The position and velocity are updated at alternating half time steps as illustrated in figure 30, just as the name suggests (the velocity is "leaping" over the position and vice versa (*Bockspringen*)).

Note: To start the scheme or to save the position and velocity at the same time, a half step needs to be performed, e.g. using half standard explicit Euler.

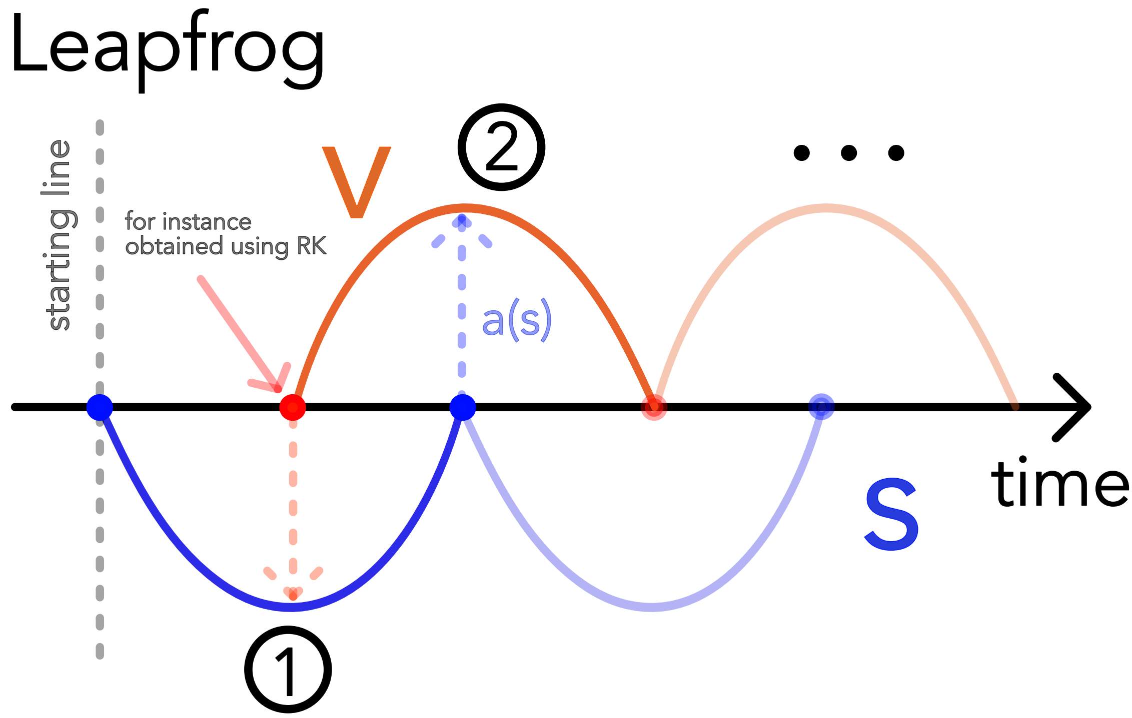


Figure 30: Illustration of the leapfrog scheme.

3.7.5.1 Connection between Leapfrog and Velocity Verlet

We can combine steps $4, 5, 1$ in the velocity Verlet scheme in the half-step formulation to get

$$\begin{aligned} \underline{s}(t+\Delta t) &= \underline{s}(t) + \underline{v}\left(t+\frac{1}{2}\Delta t\right)\Delta t \\ \underline{v}(t+\frac{3}{2}\Delta t) &= \underline{v}(t+\frac{1}{2}\Delta t) + \underline{a}(t+\Delta t)\Delta t \end{aligned} \tag{99}$$

which if we shift everything by $\frac{1}{2}\Delta t$ yields the aforementioned leapfrog scheme. Phrased differently velocity Verlet is leapfrog with Euler-kickoff built-in.

3.7.5.2 Kick-drift-kick and Drift-kick-drift Leapfrog formulations to have velocity and position information at the same time

Problem: One problem of the Leapfrog scheme in this form is that velocity and position information are not available at the same time. This is for instance problematic if we want to calculate the energy which depends on both or if we want to change the step-size adaptively without destryoing the interlacement (see figure 31).

Idea: We rearrange and split the interlaced integration by a full time step into a half step in first variable, full step in second variable, half step in first variable.

There are two possible schemes.

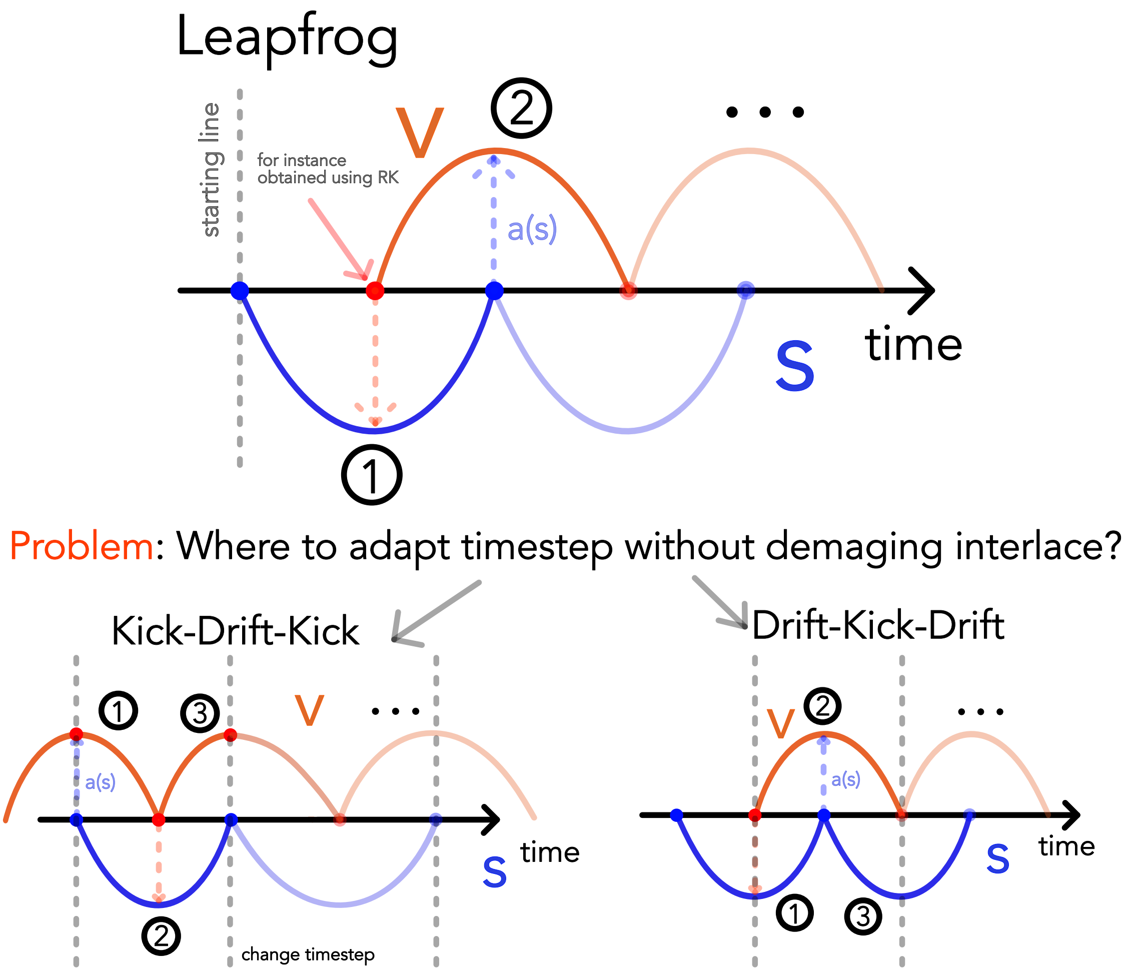


Figure 31: Illustration of the problem of changing the step-size in the leapfrog scheme and the solutions - kick-drift-kick and drift-kick-drift.

In the **kick-drift-kick formulation**, positions are stored at full, velocities at half time steps (equivalent to the velocity Verlet).

$$
\begin{aligned}
&\underline{v}(t+\frac{1}{2}\Delta t) = \underline{v}(t) + \underline{a}(t)\frac{\Delta t}{2} \quad \text{kick, half-step in first variable}\\
&\underline{s}(t+\Delta t) = \underline{s}(t) + \underline{v}(t+\frac{1}{2}\Delta t)\Delta t \quad \text{drift, full-step in second variable}\\
&\underline{v}(t+\Delta t) = \underline{v}(t+\frac{1}{2}\Delta t) + \underline{a}(t+\Delta t)\frac{\Delta t}{2} \quad \text{kick, half-step in first variable}\\
&\text{possibly adapt step-size here}\\
&t \rightarrow t+\Delta t
\end{aligned}
\tag{100}
$$

where a *drift* is a change of the position with constant velocity and a *kick* is a change of the velocity with constant acceleration.

In the **drift-kick-drift formulation**, velocities are stored at full, positions at half time steps.

$$
\begin{aligned}
&\underline{s}(t+\frac{1}{2}\Delta t) = \underline{s}(t) + \underline{v}(t)\frac{\Delta t}{2} \quad \text{drift, half-step in first variable}\\
&\underline{v}(t+\Delta t) = \underline{v}(t) + \underline{a}(t+\frac{1}{2}\Delta t)\Delta t \quad \text{kick, full-step in first variable}\\
&\underline{s}(t+\Delta t) = \underline{s}(t+\frac{1}{2}\Delta t) + \underline{v}(t+\Delta t)\frac{\Delta t}{2} \quad \text{drift, half-step in first variable}\\
&\text{possibly adapt step-size here}\\
&t \to t+\Delta t
\end{aligned}
\tag{101}
$$

3.7.5.3 Advantages of the Leapfrog scheme

The leapfrog scheme is second order accurate, symmetric (time reversible), symplectic, has good energy conservation properties and is time reversible (proofs in Springel et al., 2023, chapter 2.8).

Second order accuracy may be surprising as we seemingly only Taylor approximate up to the first order - but note the interlacement and connection to velocity Verlet.

3.7.5.4 Leapfrog is symmetric (time reversible)

In terms of the one-step map $\Phi_h : (\underline{s}_n, \underline{v}_n) \to (\underline{s}_{n+1}, \underline{v}_{n+1})$ symmetry means that $\Phi_{-h}^{-1} = \Phi_h$ so going one step forward and then back leads us back to where we came from, $n \rightleftharpoons n+1$.

We integrate from $\left(\underline{s}_n, \underline{v}_{n-\frac{1}{2}}\right)$ to $\left(\underline{s}_{n+1}, \underline{v}_{n+\frac{1}{2}}\right)$ and return.

$$
\begin{aligned}
\underline{s}_{\text{fin}} &= \underline{s}_{n+1} - \underline{v}_{n+\frac{1}{2}}\Delta t = \underline{s}_n + \underline{v}_{n+\frac{1}{2}}\Delta t - \underline{v}_{n+\frac{1}{2}}\Delta t = \underline{s}_n\\
\underline{v}_{\text{fin}} &= \underline{v}_{n+\frac{1}{2}} - \underline{a}_n\Delta t = \underline{v}_{n+\frac{1}{2}} + \underline{a}_n\Delta t - \underline{a}_n\Delta t = \underline{v}_{n-\frac{1}{2}}
\end{aligned}
\tag{102}
$$

So leapfrog is symmetric (time-reversible), different from e.g. explicit Euler (see figure 32).

Advantages of symmetric methods: Symmetric methods applied to *integrable and near-integrable* **reversible systems** share similar properties to symplectic methods applied to *(near)-integrable* Hamiltonian systems: linear error growth and long-time near-conservation of first integrals. For a non-reversible system, a symmetric but non-symplectic method (e.g. Lobatto IIIB) will have no good conservation properties though.

Note: Not every symplectic method is symmetric. For instance symplectic Euler is symplectic but not symmetric.

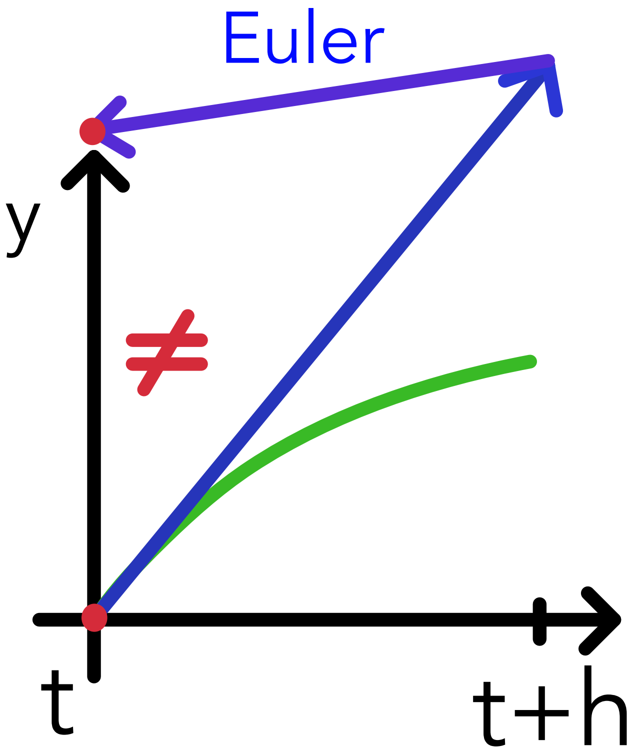


Figure 32: Explicit Euler is not symmetric (time-reversible).

3.7.5.5 Symplecticity of the leapfrog scheme I: Intuition and Meaning

Symplecticity (so area conservation in phase space) is illustrated at the hand of the pendulum in figure 33 and energy conservation at the hand of the two-body-problem in figure 34. The area in phase space does not change at all while the energy shows small fluctuations but no overall drift (compare Hairer, Lubich, et al., 2003, theorem 5.5). The angular momentum is exactly conserved in the leapfrog solution to the two-body-problem (details on the conservation of specific *quadratic first integrals* can be found in Hairer, Lubich, et al., 2003, theorem 3.5). As visible in the changing orientation of the orbit in figure 34 the leapfrog scheme does not preserve the orientation of the Runge-Lenz vector.

3.7.5.6 Symplecticity of the leapfrog scheme II: Proof

Consider the separable Hamiltonian

$$
\begin{aligned}
H(q,p) &= H_{\text{kin}}(p) + H_{\text{kin}}(q) \\
&\underbrace{=}_{\text{here}} \frac{p^2}{2m} + U(q)
\end{aligned}
\tag{103}
$$

Procedure: We solve both parts of the Hamiltonian separately (operator splitting) and construct Leapfrog as the concatenation of these solutions (proving symplecticity) and then

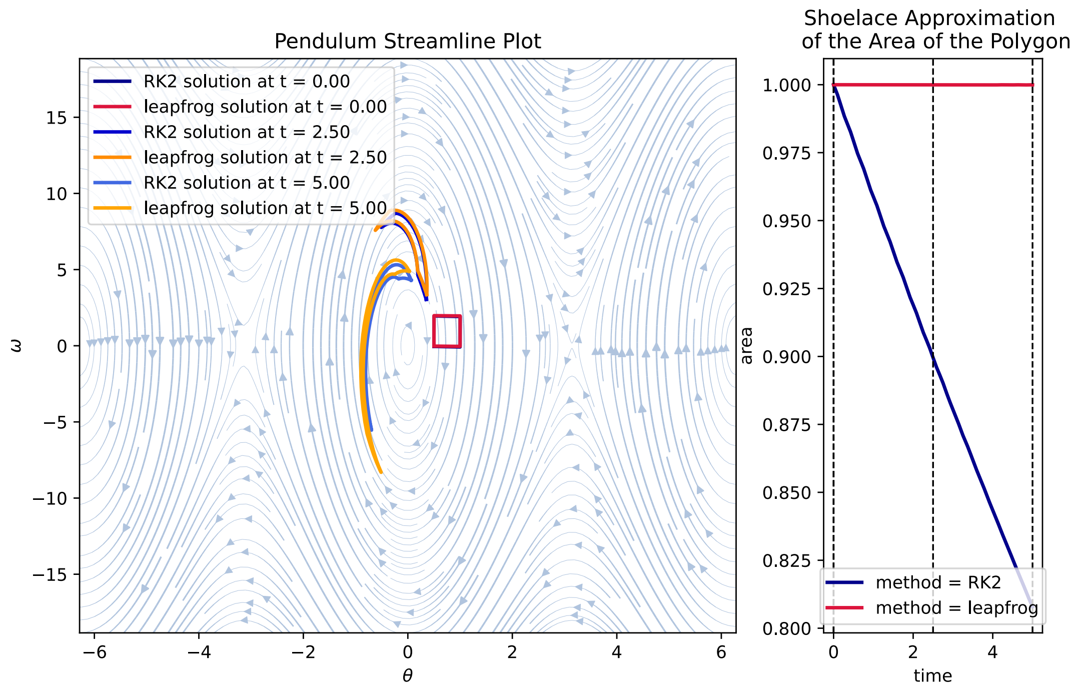


Figure 33: The same situation as in figure 28 is shown but now with the result of the leapfrog method added. The leapfrog scheme preserves the area in phase space while the RK2 scheme does not.

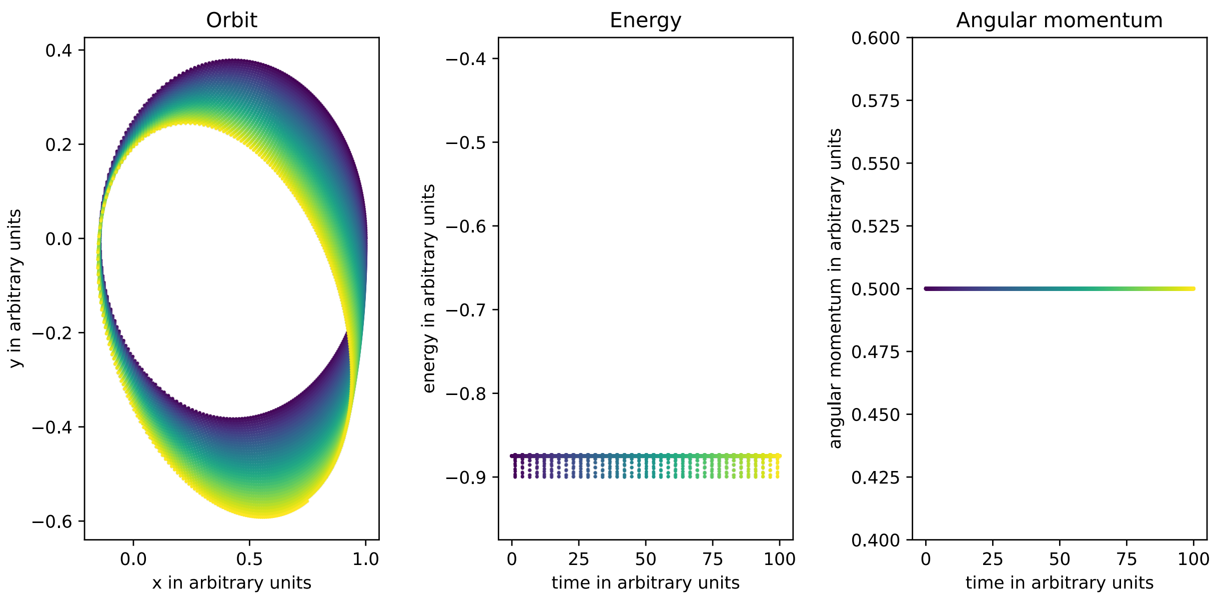


Figure 34: Numerical solution of the two-body-problem using the leapfrog method in the kick-drift-kick scheme. The left panel shows the orbit, the central one the energy and the right one the angular momentum. Time is also encoded in the form of color, going from a dark blue to yellow.

calculate an error Hamiltonian.

Operator splitting From the Hamiltonian equations of the kinetic and potential part, we find update steps.

For the kinetic part H_{kin} we find

$$\left.\begin{aligned} \partial_t q = \partial_p H_{\text{kin}} = \tfrac{p}{m} \\ \partial_t p = -\partial_q H_{\text{kin}} = 0 \end{aligned}\right\} \rightarrow \left\{\begin{aligned} q^{(n+1)} &= q^{(n)} + \tfrac{p^{(n)}}{m}\Delta t \\ p^{(n+1)} &= p^{(n)} \end{aligned}\right. \tag{104}$$

and for the potential part

$$\left.\begin{aligned} \partial_t q = \partial_p H_{\text{pot}} = 0 \\ \partial_t p = -\partial_q H_{\text{pot}} = -\partial_q U \end{aligned}\right\} \rightarrow \left\{\begin{aligned} q^{(n+1)} &= q^{(n)} \\ p^{(n+1)} &= p^{(n)} - \partial_q U \Delta t \end{aligned}\right. \tag{105}$$

Note: Independent of Δt the schemes found are exact; and symplectic as brought forward by Hamiltonians.

Constructing leapfrog Let $\phi_{\Delta t}(H)$ describe making a step Δt governed by H in phase-space (a time evolution operator). Then leapfrog (kick-drift-kick version) is given as

$$\phi_{\Delta t}(H) = \phi_{\frac{\Delta t}{2}}(H_{\text{pot}}) \odot \phi_{\frac{\Delta t}{2}}(H_{\text{kin}}) \odot \phi_{\Delta t}(H_{\text{pot}}) \tag{106}$$

so Leapfrog is symplectic as a concatenation of symplectic operators, so

- there is no secular (long-lasting, non-oscillatory) drift in e.g. the Energy of e.g. the Kepler orbits
- the longer the timespan to simulate, the more it makes sense to use leapfrog over Runge Kutta schemes (e.g. the explicit Euler scheme always overshoots the orbits leading to increasing total energy and unbound states)

Note: »Yes, symplectic integrators do not exactly conserve energy. It is a common misconception that they do. What symplectic integrators actually do is solve for a trajectory which rests on a symplectic manifold that is perturbed from the true solution's manifold by the truncation error. This means that symplectic integrators do not experience (very much) longtime drift, but their orbit is not exactly the same as the true solution in phase space, and thus you will see differences in energy that tend to look periodic. There is a small drift which grows linearly and is related to floating-point error, but this drift is much less than standard methods. This is why symplectic methods are recommended for longtime integration.« (Rackauckas, Sciemon, et al., 2022)

Error Hamiltonian Indeed, it turns out that the leapfrog scheme exactly solves the modified Hamiltonian

$$H_{leap} = H + H_{err}, \quad H_{err} \propto \frac{\Delta t^2}{12} \left\{ \{H_{kin}, H_{pot}\}, H_{kin} + \frac{1}{2} H_{pot} \right\} + \mathcal{O}(\Delta t^3) \tag{107}$$

where H_{kin} and H_{pot} are the kinetic and potential part of the original Hamiltonian (Springel et al., 2023, chapter 2.8). The curly brackets denote the Poisson bracket.

Notes on the derivation of the Error Hamiltonian The time evolution of a phase space function $F(p, q)$ under the flow generated by a Hamiltonian H fulfills $-\partial_t F = -\{H, F\} = -\hat{H} F$ and similar to the time-development operator in Quantum Mechanics, we can write

$$F(t) = \exp\left(\hat{H} t\right) F(0) \tag{108}$$

and for the leapfrog scheme with $H_{leap} = H + H_{err}, H = H_{kin} + H_{pot}$ we have

$$\exp\left((H + H_{\text{err}})\, \Delta t\right) = \exp\left(H_{\text{pot}} \frac{\Delta t}{2}\right) \exp\left(H_{\text{kin}}\, \Delta t\right) \exp\left(H_{\text{pot}} \frac{\Delta t}{2}\right) \tag{109}$$

where using the the Baker-Campbell-Hausdorff formula lets us find H_{err}.

3.8 Extrapolation method: Bulirsch-Stoer algorithm

Our goal is still to obtain highly accurate and cheap solutions to ODEs. The ingedriends of Bulirsch-Stoer are

1. we integrate across a whole interval with length H multiple times with a sequence of decreasing substep-sizes h_j each yielding a final result $F(h_j)$

2. we extrapolate $F(h)$ to $h = 0$ asking ourselves *What would be the solution, if we had taken infinitely many, infinitely small steps?*, for instance using polynomial extrapolation or Richardson extrapolation[3] with rational functions

with the method being most-appropriate for differential equations containing smooth (cheap to evaluate) functions. An illustration is given in figure 35.

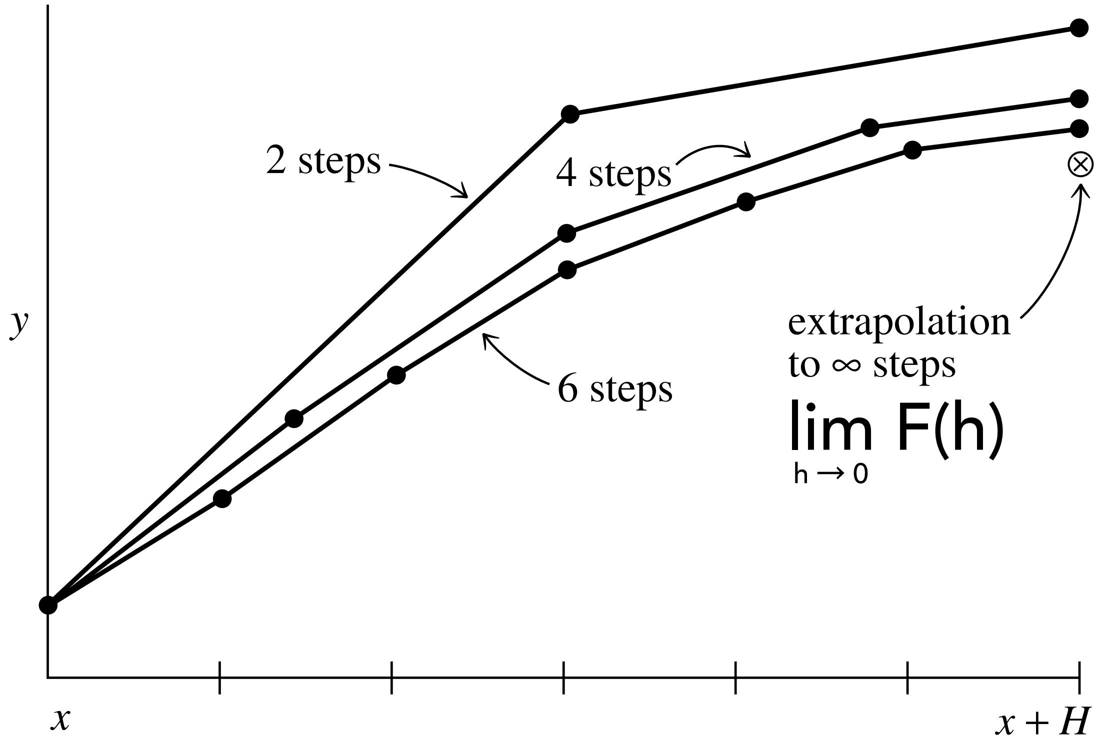


Figure 35: Illustration of the Bulirsch-Stoer algorithm.

While the idea of this method is very beautiful, its usage is disputed, with e.g. W. Van Snyder writing: extrapolation methods are almost always substantially inferior to Runge-Kutta, Taylor's series, or multistep methods.

A single step taking us from x to $x + H$ (large distance H) consists of many substeps using the modified midpoint rule.

[3]A sequence acceleration method to improve the convergence of a sequence $F^* = \lim_{h\to 0} F(h)$

3.8.1 Basic integration method | second order method with $\mathcal{O}(h^2)$; midpoint rule → modified midpoint rule

The midpoint rule is given by

$$\begin{aligned} k_1 &= f(y_i, x_i) \\ k_2 &= f\left(y_i + \frac{h}{2}k_1, x_i + \frac{h}{2}\right) \\ x_{i+1} &= x_i + h \\ y_{i+1} &= y_i + hk_2 + \mathcal{O}(h^3) \end{aligned} \tag{110}$$

as illustrated in figure 36.

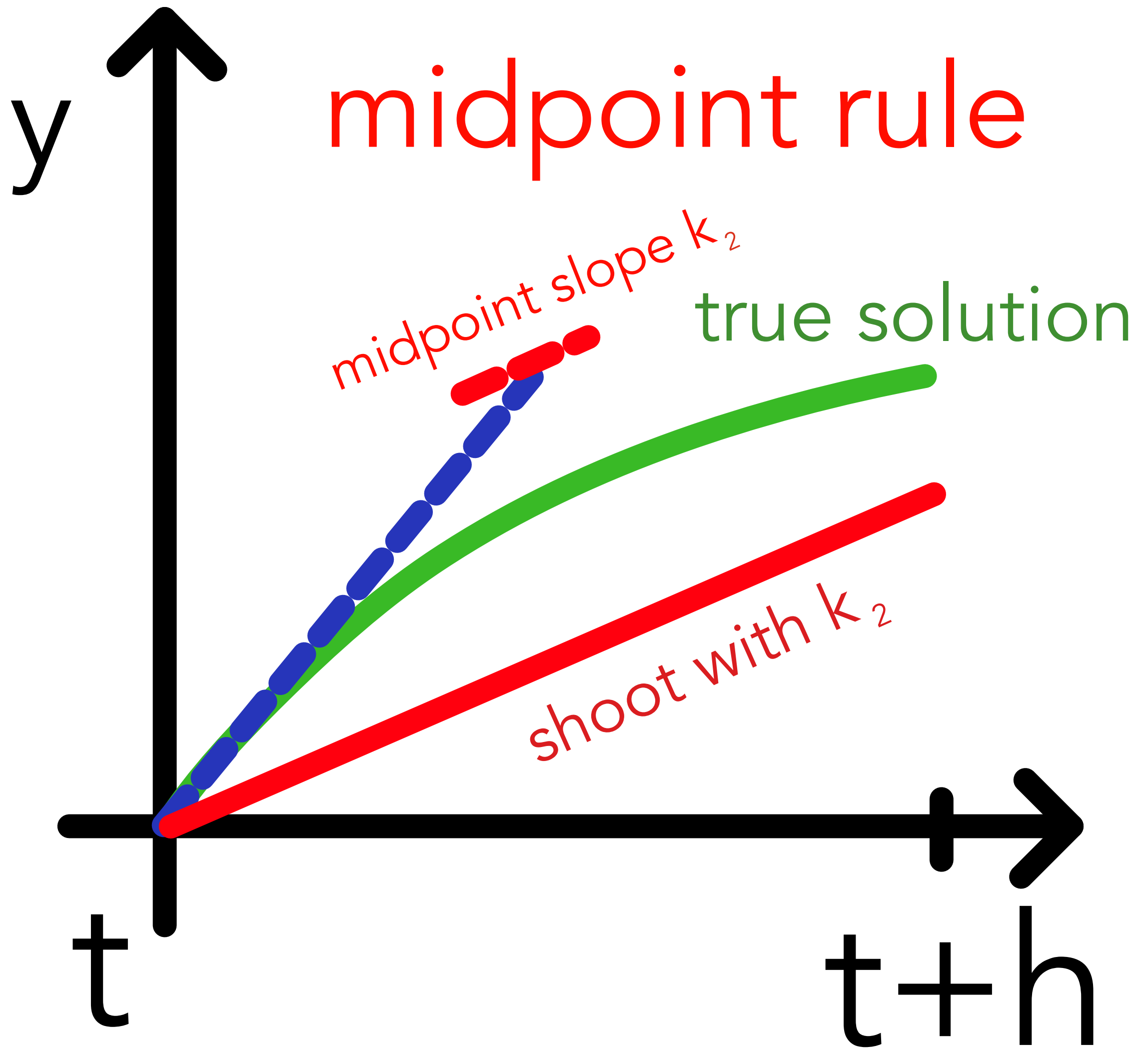


Figure 36: Illustration of the midpoint rule.

3.8.1.1 Modified midpoint rule

We advance from x to $x+H$ using n substeps of size $h = \frac{H}{n}$. Except for the first and last step, we advance using the midpoint rule.

Advantage of the midpoint rule over RK2: We only need one evaluation of f per step h instead of two.

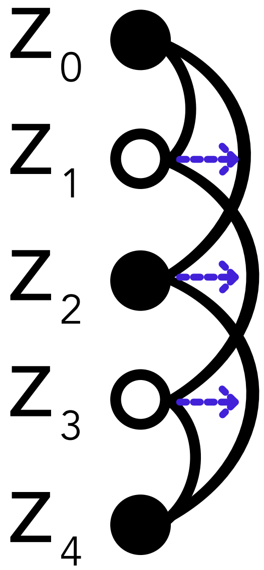


Figure 37: Modified midpoint rule.

$$
\begin{aligned}
z_0 &= y(x) \\
z_1 &= z_0 + hf(z_0, x) \text{ not midpoint} \\
z_2 &= z_0 + 2hf(z_1, x+h) \text{ midpoint with stepsize } 2h \\
z_3 &= z_1 + 2hf(z_2, x+2h) \text{ midpoint with stepsize } 2h \\
&\vdots \\
z_m &= z_{m-2} + 2hf(z_{m-1}, x+(m-1)h) \\
z_{m+1} &= z_{m-1} + 2hf(z_m, x+mh), \quad m = 4, \ldots, n-1 \\
&\vdots \\
y(x+H) \approx y_n &= \frac{1}{2}\left[z_n + \underbrace{z_{n-1} + hf(z_n, x+H)}_{\text{euler Step from } z_{n-1}} \right]
\end{aligned}
\tag{111}
$$

3.8.1.2 Combining modified midpoint calculations with different h; advantage of modified midpoint

Remember that the midpoint rule is of 2nd order so the error when covering an interval H with multiple steps is $\mathcal{O}(h^2)$. Central to the advantage of the modified midpoint rule is (not proven here)

$$
y(x+H) - y_n = \sum_{i=1}^{\infty} \alpha_i h^{2i} \tag{112}
$$

so the error between the true $y(x+H)$ and our numerical result y_n expressed as a power series in h only contains even powers of h. We can therefore combine results with different step-sizes to gain two oders at a time.

Example Let us combine a version with n steps h and one with $2n$ steps $2h$.

$$\begin{aligned} n: \quad & y(x+H) - y_n = \alpha_1 h^2 + \alpha_2 h^4 \\ 2n: \quad & y(x+H) - y_{2n} = \alpha_1 \left(\frac{h}{2}\right)^2 + \alpha_2 \left(\frac{h}{2}\right)^4 \end{aligned} \tag{113}$$

from which we obtain

$$y(x+H) = \frac{4y_{2n} - y_n}{3} + \mathcal{O}(h^4) \tag{114}$$

which is 4th order accurate like RK4 but at less cost (function evaluations).

3.8.1.3 What extrapolation nodes to choose? - how to inrease n (or rather decrease h)

Let us remember

$$F(h_n) = y_n \quad \text{with } n = \frac{H}{h} \tag{115}$$

We cross the large interval H using multiple substeps multiple times with decreasing substep size. Each iteration delivers a result $F(h_n)$ and we have already seen that such results can be combined smartly - and extrapolation to $h \to 0$ is even better. But what evalutation nodes for $F(h)$ should we choose?

$$\begin{aligned} \text{Romberg}: \quad & n = [2, 4, 8, 16, \ldots, 1024] \\ \text{Bulirsch}: \quad & n = [2, 4, 6, 8, 12, 16, \ldots, 96] \\ \text{Deufelhard}: \quad & n = [2, 4, 6, 8, 10, \ldots, 24] \end{aligned} \tag{116}$$

3.8.1.4 How to extrapolate from multiple $F(h_n)$ to the limit $h \to 0$?

One option is polynomial interpolation (two points define a line, ..., n points define a polynomial of (max) degree $n-1$), so we do polynomial regression and evaluate it at zero.

There is also the approach to use rational functions, which can also capture poles and divergence regions between the interpolation points (which polynomials will never do), so

$$R_{m+1}(x) = \frac{P_\mu(x)}{Q_\nu(x)} = \frac{p_0 + p_1 x + p_2 x^2 + \cdots + p_\mu x^\mu}{q_0 + q_1 x + q_2 x^2 + \cdots + q_v x^v}, \quad m + 1 = v + \mu + 1 \tag{117}$$

Hairer, Wanner, and Nørsett, 1993 write: »Many authors in the sixties claimed that it is better to use rational functions instead of polynomials. Later numerical experiments (Deuflhard 1983) showed that rational extrapolation is nearly never more advantageous than polynomial extrapolation.«. One reason I can imagine is that the more complex rational model is more unstable and harder to appropriately fit.

3.9 Predictor-corrector methods

Multistep idea: While a one-step method only uses the differential value and the ODE itself in a step, multistep methods also include information from previous steps (e.g $x, x-h, x-2h$) to obtain better estimates for the next step $(x+h)$. The method is e.g. kicked off using Runge-Kutta steps.

Let us start by writing an advance in an ODE $\partial_x y = f(y(x), x)$ as

$$y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} f(x', y(x'))dx' \tag{118}$$

the problem is that different from an integration problem, we would need to know $y(x)$ to use this formula to calculate $y(x)$ or rather y_{n+1} - so what have we won?

Consider we are in the multistep setting and already have approximations $y_n, y_{n-1}, \ldots$ at $x_n, x_{n-1}, \ldots$. We can then approximate $f(x, y)$ by a polynomial passing through those points and yield

$$y_{n+1} = y_n + h \cdot (\beta_0 f(x_{n+1}, y_{n+1}) + \beta_1 f(x_n, y_n) + \beta_2 f(x_{n-1}, y_{n-1}) + \ldots) \tag{119}$$

But wait - we used y_{n+1} in the RHS which we do not know. We could get an explicit scheme by setting $\beta_0 = 0$ but the formula screams fixpoint iteration (as in Picard iteration) - **predictor-corrector method**.

1. **predictor step:** obtain a good estimate for y_{n+1}
2. **corrector step(s):** plugging the result of the predictor step into eq. 119 gives an improved estimate for y_{n+1}

For correction to make sense, the first prediction must be sufficiently good.

3.9.1 One-step predictor-corrector method: RK2 and $P(EC)^k$

Indeed standard RK2 is a predictor-corrector method

$$\begin{aligned} k_1 &= f\left(y_n, x_n\right) \\ k_2 &= f\left(\underbrace{y_n + hk_1}_{\text{predictor}}, x_n + h\right) \\ &\underbrace{y_{n+1} = y_n + \frac{h}{2}\left(k_1 + k_2\right) + \mathcal{O}\left(h^3\right)}_{\text{corrector}} \end{aligned} \tag{120}$$

Different notation and $P(EC)^k$ method We write the predictor P step as

$$\tilde{y}^{P}_{n+1,[0]} = y_n + hf(y_n, x_n) \tag{121}$$

and can then write the evaluation / corrector step EC as

$$\tilde{y}^{EC}_{n+1,[1]} = y_n + \frac{h}{2}\left[f(y_n, x_n) + f(\tilde{y}^{P}_{n+1,[0]}, x_{n+1})\right] \tag{122}$$

which we can repeat

$$\begin{aligned} \tilde{y}^{EC}_{n+1,[2]} &= y_n + \frac{h}{2}\left[f(y_n, x_n) + f(\tilde{y}^{EC}_{n+1,[1]}, x_{n+1})\right] \\ \tilde{y}^{EC}_{n+1,[k]} &= y_n + \frac{h}{2}\left[f(y_n, x_n) + f(\tilde{y}^{EC}_{n+1,[k-1]}, x_{n+1})\right] \end{aligned} \tag{123}$$

until convergence $|\tilde{y}^{EC}_{n+1,[k]} - \tilde{y}^{EC}_{n+1,[k-1]}| \leq \epsilon_0$ (some error tolerance ϵ_0) where our final approximation is $y_{n+1} = \tilde{y}^{EC}_{n+1,[k]}$. This is the $P(EC)^k$ method.

3.9.2 4th order Adams-Bashforth-Moulton

Here we used the multistep principle as introduces above. In the 4th order Adams-Bachforth-Moulton approach we use three previous timesteps. It is a 4th order method.

The predictor step has weights designed so it gives the correct result for cubic polynomials

$$\tilde{y}_{n+1} = y_n + \frac{h}{24}\left[55f(y_n, x_n) - 59f(y_{n-1}, x_{n-1}) + 37f(y_{n-2}, x_{n-2}) - 9f(y_{n-3}, x_{n-3})\right] + \mathcal{O}(h^5) \tag{124}$$

the evaluation / corrector step EC is

$$y_{n+1} = y_n + \frac{h}{24}\left[9f(\tilde{y}_{n+1}, x_{n+1}) + 19f(y_n, x_n) - 5f(y_{n-1}, x_{n-1}) + f(y_{n-2}, x_{n-2})\right] + \mathcal{O}(h^5) \tag{125}$$

containing $\tilde{y}_{n+1}$ (*implicit*) and can be repeated for higher accuracy (plug in y_{n+1} instead of $\tilde{y}_{n+1}$ in the next EC step). We start the scheme with three RK steps.

3.10 Shooting | adapting parameters until boundary conditions are fulfilled

3.10.1 Remark on ODE solutions in phase space

There are infinitely many solutions to an ODE but only one unique to a initial value problem. Those solutions are streamlines through phase space that

- do not start / end
- do not cross

3.10.2 Examplary Shooting Problem

Consider the motion of a projectile from a canon, given by

$$\partial_t^2 \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ -g \end{pmatrix} - b\partial_t \begin{pmatrix} x \\ y \end{pmatrix} \tag{126}$$

where the canon sits at $(0, 1)$ and shoots with a give velocity v_{canon} at an angle α to the horizontal. Our aim is hitting a target at $(x_{\text{target}}, y_{\text{target}})$ (boundary condition). This is illustrated in figure 38.

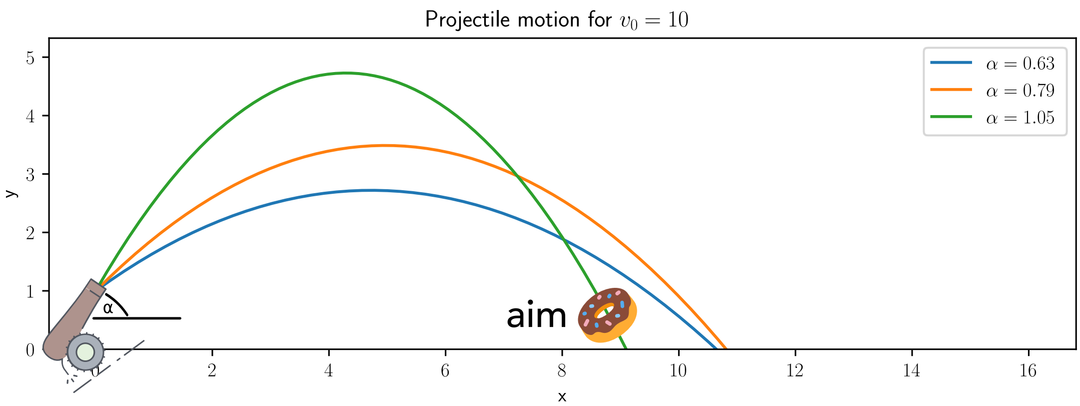


Figure 38: Illustration of the shooting problem. Trajectories gained by converting the system to first order ($\underline{q} = (x, y, v_x, v_y)$, $f = (v_x, v_y, -bv_x, -g - bv_y)$) and then applying RK2.

3.10.3 Shooting

In the shooting method a boundary value problem is reduced to an initial value problem. We solve the initial value problem for different choices of parameters until the given boundary conditions are satisfied – we shoot trajectories in different directions from one boundary until we find one that hits the other boundary condition. For a neutron star we could know the central density and density at the surface and seek some process parameters.

4 Simulation of Physical Systems - from Quantum Mechanics to Fluid Dynamics

4.1 Different levels of modelling from Quantum Mechanics to Kinetic Gas theory

Our aim is simulating real-world physical systems with two of our fundamental tools being modeling and model order reduction. When we model a fluid, we will probably not model every particle let alone the underlying quantum mechanics[4].

Starting off with Quantum Mechanics Consider we would model the whole wave function of a system of particles $\Phi(\underline{x}_1, \ldots, \underline{x}_N)$ (giving the joint probability $|\Phi|^2$ of finding all particles at $\underline{x}_1, \ldots, \underline{x}_N$ with single particle probabilities being the margins) with possibly more degrees of freedom (like spin) and evolve it using e.g. the Schrödinger equation with particle interaction being represented in the potential of the Hamiltonian. If we would discretize each dimension with 1000 cells, the simulation would have 1000^{3N} cells - quickly infeasible (we model a too large probability space). First reduction steps could be

- use symmetries, e.g. particles of a type are indistinguishable reducing the degrees of freedom
- in given potentials approximate the particle wave as a sum of standing waves and evolve the coefficients
- ...

Molecular dynamics Quantum mechanical simulation is infeasible (and unnecessary) for larger systems. A first step is to decouple the electrons and protons (Born-Oppenheimer approximation) as the electrons move on much quicker timescales and describe the atoms as localized particles in phase-space. Based on the positions of the nuclei forming a fixed potential, we can calculate the lowest energy wave function of the electrons (which will occur in *no time* for the protons), from there forces on the nuclei with which we move them and so on. The next step is to model the interatomic interactions using inter-atomic potentials (e.g. Morse, Lenard-Jones) with the potential parameters for instance adapted to match quantum mechanical simulations or for a system as a whole to portray correct behavior.

[4]Except for instance when we want to analyse shocks in Warm Dense Matter in Neutron Stars or inertial confinement fusion based on Quantum hydrodynamics (Graziani et al., 2022)

Kinetic Gas theory Let us assume, the "free flight" between interactions is much larger than the interaction time. We can model *molecule-particles* with some effective interaction radius which only exhibit collisions. Note that while when modeling the potentials, at a sufficiently large collision parameter, particles do a kind of slingshot maneuver (attractive potential), which can never be modelled in a pure collision system. But this can be fine - with the right parameters our system can work globally in spite of local differences.

4.2 From a classical particle description to the Boltzmann equation

Consider we start of with classical point-like particles in phase space (e.g. the molecules) $\{\underline{x}_i, \underline{p}_i\}_{i=1}^N$ with a force term

$$\frac{d}{dt}\underline{p}_i = \underline{f}(\underline{x}_i(t), t) \underbrace{=}_{\substack{\text{e.g.}\\ \text{for}\\ \text{ions}}} q_i \cdot \left(\underline{E}_m(\underline{x}_i(t), t) + \frac{1}{m_i}\underline{p}_i(t) \times \underline{B}_m(\underline{x}_i(t), t) \right) \tag{127}$$

where the forces depend on the particle positions and momenta themselves (e.g. via Maxwells equations where the particle positions inform the charge density and thus the electric field and the velocities the currents and thus the magnetic field).

For as many particles as in a fluid (a mol has $\sim 6 \cdot 10^{23}$ particles and e.g. water has roughly 18 grams per mol) this is still unfeasible for simulation. It turns out to be smarter to model a phase space density. The exact classical phase space density is

$$\mathcal{F}(\underline{x}, \underline{p}, t) = \sum_{i=1}^{N} \delta(\underline{x} - \underline{x}_i)\delta(\underline{p} - \underline{p}_i) \tag{128}$$

Phase space conservation (Liouville theorem) (previously visualized for the pendulum) means that

$$\frac{d}{dt}\mathcal{F}(\underline{x}, \underline{u}, t) = \partial_t \mathcal{F} + \underline{v}\underline{\nabla}_{\underline{x}}\mathcal{F} + \underline{\mathcal{A}}\underline{\nabla}_{\underline{u}}\mathcal{F} = 0 \tag{129}$$

with the local acceleration $\underline{\mathcal{A}}$ following from the local force and mass. Let us use a mean phase space density and acceleration instead.

$$\begin{aligned} \mathcal{F}(\underline{x}, \underline{u}, t) &= f(\underline{x}, \underline{p}, t) + \delta\mathcal{F}(\underline{x}, \underline{u}, t), \quad & f(\underline{x}, \underline{u}, t) &= \langle \mathcal{F}(\underline{x}, \underline{u}, t) \rangle \\ \underline{\mathcal{A}}(\underline{x}, \underline{u}, t) &= \underline{a}(\underline{x}, \underline{u}, t) + \delta\underline{\mathcal{A}}(\underline{x}, \underline{u}, t), \quad & \underline{a}(\underline{x}, \underline{u}, t) &= \langle \underline{\mathcal{A}}(\underline{x}, \underline{u}, t) \rangle \end{aligned} \tag{130}$$

which we have also done for the acceleration, separating a mean effect on a *fluid parcel* from the direct particle-particle interactions. With this we get

$$\partial_t f + \underline{v} \cdot \underline{\nabla}_{\underline{x}} f + \underline{a} \cdot \underline{\nabla}_{\underline{u}} f = -\langle \delta \underline{\mathcal{A}} \cdot \underline{\nabla}_{\underline{u}} \delta \mathcal{F} \rangle =: \left. \frac{df}{dt} \right|_c \tag{131}$$

which is the Boltzmann equation, where we identified the local fluctuations with collisions $\frac{df}{dt}\big|_c$ from a kinetic gas theory perspective.

4.3 Emergence of irreversibility in the Boltzmann equation

Consider a classical simulation of colliding spheres (interaction time $\ll$ free flight time), where we start out with the particles concentrated at the center of our box. As of their thermal motion, the particles will spread out and fill the box. Now consider we start with this end state and reverse all velocities - the particles will clump up - anti-dissipation. Such things can happen miscroscopically, they are simply very unlikely.

Something unlikely miscroscopically is virtually impossible macroscopically.

But based on the Boltzmmann equation, this will never happen - the Boltzmann equation is *irreversible* and will never show anti-dissipation.

»The derivation of the Boltzmann equation (BE) from the Hamiltonian equations of motion of a hard spheres gas is a key topic on irreversibility (Sklar 1993, p.32; Uffink 2007, Section 4). Although the Hamiltonian equations of motion are invariant under time reversal, the BE is not. Moreover, this equation allows us to derive the H-theorem, which states that a function H monotonically decreases with time, and thus, that the minus-H function increases, in agreement with the second law of thermodynamics. The derivation of the BE thus raises the question of irreversibility, since this equation exhibits irreversibility even though the microscopic description of the gas is based on reversible equations.« from Ardourel, 2017 where the emergence of irreversibility is discussed in detail.

We can try to understand the emergence of irreversibility based on going from the exact information of the positions and momenta of the particles to an averaged phase space density:

> »In other words, when the particles are described by the Boltzmann equation, our knowledge is incomplete, since the positions and momenta of all the particles remain unknown, in contrast to the description by means of Hamilton canonical equations or Liouville equation. And this lack of knowledge makes the evolution of the one-particle distribution function f irreversible. The irreversibility is then explicitly expressed by the collision integral in the Boltzmann equation. The second law of thermodynamics thus emerges from completely reversible dynamics when our description is incomplete (not seeing all positions and momenta of the particles).« from Kincl and Pavelka, 2023

5 Basic Fluid Dynamics

Fluids (gases or liquids[5]) react to tangential (aka shear) stress with flow, a deformation rate depending on the viscosity, as opposed to solids which deform. Many systems from galaxies to lab plasmas can - on the right scale - be successfully modelled as fluids.

5.1 Basic notes on fluid description - the fluid from the view of a parcel - macroscopic fluid view

Physical systems can be described on different level: from a wave function in quantum physics, over a collection of point-like particles in classical physics to a **continuous fluid**.

5.1.1 When is a fluid description valid?

We want to describe the fluid in terms of macroscopic position and time dependent quantities like: mass density ρ, temperature T, velocity $\underline{u} = \underline{v} + \underline{w}$ where $\underline{v} = \langle \underline{u} \rangle$ is the mean (bulk) velocity of the local fluid element and $\underline{w}$ is the random velocity defining the temperature.

5.1.1.1 Connection between temperature and random movement

As of the equipartition theorem, each exitable degree of freedom adds $\frac{k_B T}{2}$ to the internal energy, so

$$\langle \frac{1}{2} m w_i^2 \rangle = \frac{1}{2} k_B T \quad \rightarrow \quad \langle ||\underline{w}||^2 \rangle = \frac{3 k_B T}{m} \tag{132}$$

5.1.1.2 Continuum Hypothesis

Q: When does it make sense to describe a physical system by a continuous fluid? A: When we can construct a volume small compared to the region of the fluid but big compared to the molecular free path. When we consider a fluid in terms of **parcels** with constant mass and particle number

$$\text{parcel mass } m_p = \int_V \rho dV, \quad \text{parcel volume } V, \quad \text{parcel density } \rho \tag{133}$$

we postulate that every fluid quantity (bulk quantity of the parcel) tends to a limit as the size of the volume approaches zero, before molecular fluctuation kicks in (**continuum hypothesis**), see figure 39.

[5] In gases, time between interactions is so much longer than time of interactions that they can often be described by kinetic gas theory. At higher temperatures there are more collsions and the gas is more viscous while in liquids at higher temperatures *bonds are easier to break* so the liquid is less viscous.

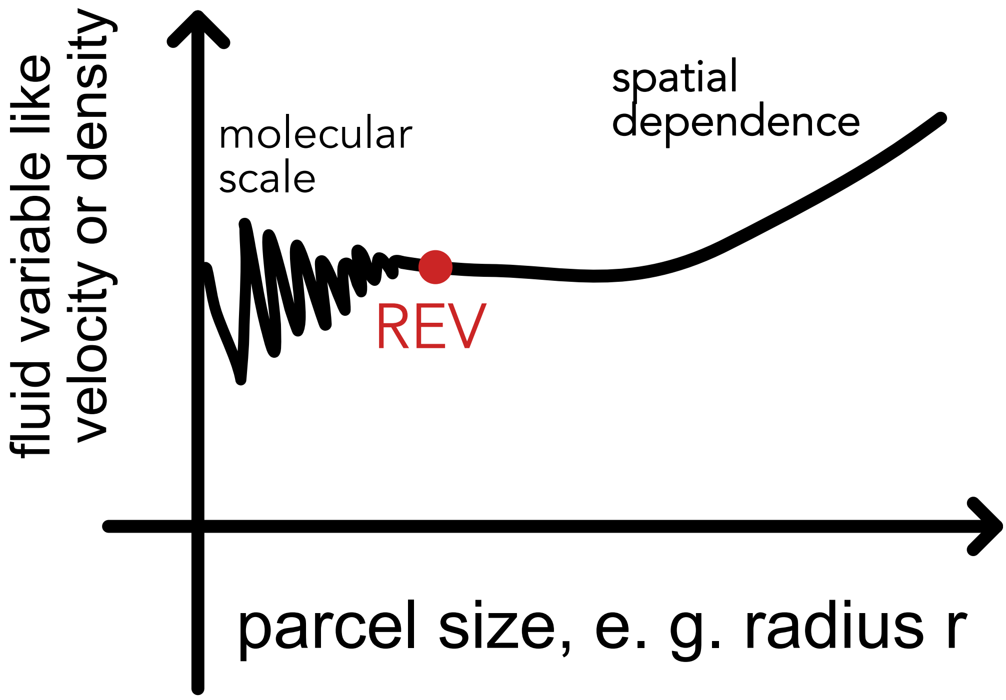


Figure 39: Continuum hypothesis; REV = representative elementary volume

A fluid description is thus justified if

$$\text{Knudsen number } Kn = \frac{\text{mean free path of particles } \lambda_{mfp}}{\text{characteristic system length } L} \ll 1 \tag{134}$$

5.1.2 Example: the plasma in the intercluster medium can be considered a fluid*

5.1.2.1 Mean free path in a model of colliding spheres

Consider a particle has a collisional cross-section of σ (in m^2) and moves with root mean square velocity $\bar{v}$ in a medium with number density n (in m^{-3}). Note that the mean relative velocity between particles is larger than the mean velocity per particle

$$\langle v_{rel}^2 \rangle = \langle (v - v')^2 \rangle = \langle v^2 \rangle - \underbrace{2\langle vv' \rangle}_{\substack{=0, \\ \text{as} \\ \text{uncor-} \\ \text{related}}} + \langle v'^2 \rangle = 2\langle v'^2 \rangle = 2\bar{v}^2 \quad \to \quad v_{rel} = \sqrt{2}\bar{v} \tag{135}$$

A particle thus in a time τ probes the volume $V = \sigma v_{rel} \tau$, so interacts with $N = nV$ particles. We are interested in the time till one interaction so the case $N = 1$, so $\tau = \frac{1}{n v_{rel} \sigma}$.

The path the particle itself has moved during that time is

$$\lambda_{mfp} = \tau\bar{v} = \frac{1}{\sqrt{2}n\sigma} \tag{136}$$

Note: The electrons though are much more mobile than the protons ($\frac{m_e}{m_p} \approx \frac{1}{1836}, m_e \approx 10^{-30}$ kg) so for electron-ion collisions the simpler $\lambda_{mfp} = \frac{1}{n\sigma}$ (assuming stationary nuclei) is probably better.

5.1.2.2 Collisional cross-section of an electron in a plasma and first approximation of λ_{mfp}

We approximate $\sigma = \pi r_{eff,e}^2$. We can approximate this radius by the distance where the electrostatic potential (in the electron-ion or electron-electron interaction) is only as relevant as the thermal energy, so (Z for the nucleic charge, relevant for the ion-electron interaction)

$$\frac{Ze^2}{r_{eff,e}} \sim m_e w_e^2 \sim k_B T_e \tag{137}$$

We can therefore approximate an electron mean free path

$$\begin{aligned} \lambda_{mfp,e} &\simeq \frac{1}{n\sigma} \sim \frac{1}{n\pi r_{eff,e}^2} \sim \frac{1}{\pi n}\left(\frac{k_B T_e}{Ze^2}\right) \\ &\sim 1.5 \times 10^{22} \left(\frac{n}{10^{-3}\,\mathrm{cm}^{-3}}\right)^{-1} \left(\frac{k_B T_e}{1\,\mathrm{keV}}\right) \mathrm{cm} \end{aligned} \tag{138}$$

where we assumed $Z = 1$.

Note: Most *collisions* in (strongly ionized) plasmas are Coulomb interactions with very small deflections (so collision parameters mostly larger than $r_{eff,e}$). Therefore, we define the collision rate and collisional cross-section based on a total deflection of $\frac{\pi}{2}$.

5.1.2.3 A better approximation to the mean free path in an ionized plasma

More careful considerations take into account the smaller deflections. The integral over $\frac{1}{b}$ (impact parameter b) yields the Coulomb logarithm

$$\log \Lambda = \log \frac{b_{max}}{b_{min}} \tag{139}$$

where for b_{max} we take the Debye length

$$\lambda_{D\xi} = \left(\frac{\epsilon_0 k_B T_\xi}{n_\xi e^2}\right)^{\frac{1}{2}}, \quad \xi = \text{ion, electron}, \quad \text{in plasma } n_i \approx n_e \tag{140}$$

(where the potential of a *Überschussladung* drops to $\frac{1}{e}\phi_0$ (e here Euler's number)) and for b_{min} e.g. $\pi r_{eff,e}$. One finally gets

$$\lambda_{mfp,e} = \frac{\bar{v}_e}{\nu_{ei}} \approx 64\pi\lambda_D \frac{\Lambda}{\log\Lambda} \propto \frac{T_e^2}{n_e} \gg \lambda_D \tag{141}$$

with ν_{ei} being the electron-ion collision rate. Examples of mean free paths are given in table 5.

Plasma	Solar Wind ($n \sim 1\,\mathrm{cm}^{-3}, T =$ some eV)	Ionosphere	Gas molecule in air under standard conditions
mean free path λ_{mfp}	1 AU	1 km	68 nm

Table 5: Mean free paths

The plasma in the intercluster medium can be considered a fluid. The solar wind can hardly be considered a fluid on a reasonable scale.

5.1.3 Fluid description based on fluid parcels

The modes of motion in a fluid are

- motion of macroscopic fluid parcels
- random walk between fluid parcels (diffusion, very slow)

The most important fluid equation, the **Navier Stokes equation** *falls out of the* Boltzmann equation but can also be *understood* as an equation of movement for a fluid parcel.

- mass conservation $\Longleftrightarrow$ continuity equation
- mass conservation + momentum conservation $\Longleftrightarrow$ Navier Stokes equation

5.1.3.1 Eulerian and Lagrangian fluid dynamics

Lagrangian and Eulerian view are presented in table 6.

The rate of change in the Eulerian view is the **material derivative**, the rate of change of a physical quantity (like temperature) of a material element that is subjected to a macroscopic

Lagrangian description	Eulerian description
Coordinate system comoves with the fluid element (we sit on the fluid parcel)	Coordinate system is fixed in space
The temporal evolution of fluid quantities is described along the trajectory of the movement of a fluid parcel	The temporal evolution of fluid quantities is described at a fixed locations over accounting volumes
e.g. measurement on a weather balloon following the wind flow	e.g. measurement by ground stations

Table 6: Lagrangian and Eulerian fluid dynamics

velocity field $\underline{v}$. Let $\Psi_L(t)$ describe the evolution of a fluid quantity along the parcel motion (Lagrangian) and $\Psi_E(\underline{x}, t)$ the evolution of a fluid quantity at a fixed location (Eulerian). Then the material derivative is

$$D_t \Psi_L = \frac{D\Psi_L}{Dt} = \partial_t \Psi_L = \underbrace{\partial_t \Psi_E}_{\substack{\text{local}\\ \text{change}}} + \underbrace{\underline{v} \cdot \underline{\nabla}_{\underline{x}} \Psi_E}_{\substack{\text{convective}\\ \text{change}}} = D_t \Psi_E \tag{142}$$

(derive $\Psi(\underline{x}(t), t)$ with respect to time and use the chain rule). Generally

$$D_t = \partial_t + \underline{v} \cdot \underline{\nabla} \tag{143}$$

5.1.3.2 Continuity equation

As of mass conservation, the change of mass m_V in a volume V can only be due to flux $\underline{j} = \rho \underline{v}$ through the surface ∂V.

$$\partial_t m_V = \partial_t \int_V \rho dV = -\int_{\partial V} \underline{j} \cdot d\underline{A} \underbrace{=}_{\text{Gauss}} -\int_V \underline{\nabla} \cdot \underline{j} dV \quad \rightarrow \quad \boxed{\partial_t \rho + \underline{\nabla} \cdot (\rho \underline{v}) = 0} \tag{144}$$

which is the continuity equation.

5.1.3.3 Incompressible fluids

An incompressible fluid is one, where the density of a fluid parcel never changes with time $\frac{d\rho}{dt} = 0$ (there can still be gradients in the density). From the Lagrangian form of the continuity equation

$$0 = \frac{d\rho}{dt} = \partial_t \rho + \underline{v} \cdot \underline{\nabla} \rho \underbrace{=}_{\substack{\text{cont.}\\ \text{eq.}}} \quad \rightarrow \quad \boxed{\underline{\nabla} \cdot \underline{v} = 0} \tag{145}$$

For $\underline{\nabla} \cdot \underline{v} < 0$ we have a sink, for $\underline{\nabla} \cdot \underline{v} > 0$ a source. The divergence of the velocity field of incompressible fluids vanishes (divergence free flow) (e. g. rotational flow) Note in a 3D flow horizontal convergence can be balanced by vertical divergence.

Interpretation of $\underline{\nabla} \cdot \underline{v} = 0$ A divergence free flow does not mean that the velocity does not change over space as one can easily see from flow along a narrowing tube. Consider a tube with flow tangential to it.

$$\begin{aligned} \underline{\nabla} \cdot \underline{v} = 0 \xrightarrow{\text{integrate over tube}} 0 = \int_V \underline{\nabla} \cdot \underline{v} dV \underset{\text{Gauss}}{=} \int_{\partial V} \underline{v} d\underline{A} \\ \xrightarrow[\text{only opposing flux trough } A_1 \text{ and } A_2]{} v_1 A_1 - v_2 A_2 = 0 \\ \rightarrow v_1 A_1 = v_2 A_2 \end{aligned} \tag{146}$$

5.1.3.4 Equation of motion of a fluid parcel, general path towards Navier-Stokes

Consider a fluid parcel with constant mass m. The equation of motion is

$$\underline{F} = \frac{d\underline{p}}{dt} = m\frac{d\underline{V}}{dt} = \rho V \frac{d\underline{v}}{dt}, \quad \underline{a} = \frac{\underline{F}}{\rho V} = \frac{\underline{f}}{\rho} = \frac{d\underline{v}}{dt} \tag{147}$$

Using the material derivative, we can write

$$\frac{d\underline{v}}{dt} = \partial_t \underline{v} + \underbrace{\underline{v} \cdot \underline{\nabla v}}_{\text{non-linear advective term} \rightarrow \text{chaotic behavior}} = \underline{a} = \frac{\underline{f}}{\rho} \tag{148}$$

where one can find (leading to a simplified **Navier Stokes equation**)

$$\underline{f} = \underbrace{\underline{f}_g}_{\text{gravi. force}} + \underbrace{\underline{f}_p}_{\text{pressure force}} + \underbrace{\underline{f}_f}_{\text{friction force}} = \rho \underline{g} - \underline{\nabla} p + \rho \nu \underline{\nabla}^2 \underline{v} \tag{149}$$

where the viscous friction $\underline{f}_v$ (viscosity ν) is an approximation for an incompressible isotropic Newtonian fluid. While pressure is a force per area, only when there are different pressures acting on the sides of a fluid parcel (a gradient), there is net movement. The friction term can be understood as diffusion of momentum when there is a velocity gradient (which only leads to a local change when when $\underline{\nabla}^2 \underline{v} \neq 0$, otherwise there is the same momentum diffusion in and out of the parcel).

5.2 Basic Gas Dynamics

Aim: Our aim is to derive equations for the evolution of variables of the fluid, like density, velocity and temperature. For instance how do perturbation (e.g. by a jet) propagate through the fluid?

5.2.1 Distribution function and Boltzmann equation

Idea: Let us derive the fluid equations based on the Boltzmann equation for the distribution function.

The distribution function $f(\underline{x}, \underline{u}, t)$ is defined so that

$$f(\underline{x}, \underline{u}, t) d^3x d^3u \tag{150}$$

is (depending on the normalization) the probability of finding a particle in the phase space volume $d^3x d^3u$ around $(\underline{x}, \underline{u})$ at time t or the expected number or particles therein, so in total

$$N = \int\int f(\underline{x}, \underline{u}, t) d^3x d^3u, \quad \text{total number of particles } N \tag{151}$$

Phase space conservation (Liouville theorem) (particles are neither created nor destroyed) is captured in the Boltzmann equation

$$\frac{d}{dt} f(\underline{x}, \underline{u}, t) = \partial_t f + \underline{\dot{x}} \underline{\nabla}_{\underline{x}} f + \underline{\dot{u}} \underline{\nabla}_{\underline{u}} f = \left. \frac{df}{dt} \right|_c, \quad \text{with } \underline{\dot{x}} = \underline{u}, \underline{\dot{u}} = \underline{g} \tag{152}$$

where $\left.\frac{df}{dt}\right|_c$ is the change in f due to *collisions*.

Note: One can understand this in the sense of dicontinupus motion (instantaneous velocity changes) kicking a particle out of a phase space volume. More aligned with our previous derivation, we can see this as a simplification of the correlation term between forces and accelerations (also capturing the particle interaction) (see subsection 4.2).

In the case of a sufficiently large collision term $\left.\frac{df}{dt}\right|_c$ (in the fluid limit $\lambda_m fp \ll L$) $f(\underline{u})$ becomes Mawellian. Near the sun for instance Coulomb collisions are incapable of isotropizing the velocity distribution of the ions in the solar wind (cigar shape).

5.2.2 Retrieving information from the Boltzmann equation

By taking the moments of the distribution function over velocity space

$$M_n(\underline{x}, t) = \int \underline{v}^n f(\underline{x}, \underline{u}, t) d^3u \tag{153}$$

we find out on the quantities of interest in space as

- density (0th moment)

$$\begin{gathered} n = \int f(\underline{x}, \underline{u}, t) d^3u, \quad \rho = nm, \quad \text{particle mass } m \\ \rightarrow \text{mass weighted average } \langle q \rangle(\underline{x}, t) = \frac{1}{\rho(\underline{x})} \int q(\underline{x}, \underline{u}, t) f(\underline{x}, \underline{u}, t) d^3u \end{gathered} \tag{154}$$

- mean velocity (1st moment)
- pressure tensor as of the fluctuations of the velocity around the mean ($\rightarrow$ temperature in case of a Maxwell distribution) (2nd moment)
- heat tensor (3rd moment)

The development of those quantities in time is given by the balance (/conservation) equations obtained from taking the appropriate moments of the Boltzmmann equation (mass, momentum and energy are conserved).

$$\text{form of balance equations: } \partial_t \text{conserved quantity} + \underline{\nabla} \text{corrsp. flux} = \text{source term} \tag{155}$$

5.2.3 Mass conservation | continuity equation (1st moment)

By following the steps

- multiply the Boltzmann equation by m and integrate over d^3u
- using Gauss divergence theorem and assuming $f \rightarrow 0$ for $u \rightarrow \infty$
- local mass conservation $\int m \left.\frac{df}{dt}\right|_c d^3u = 0$ (collisions only shift discontinuously on velocity space)

one follows the mass continuity equation

$$\boxed{\partial_t \rho + \underline{\nabla} \cdot (\rho \cdot \underline{v}) = 0, \quad \underline{u} = \underline{v} + \underline{w}, \quad \underline{v} = \langle \underline{u} \rangle \tag{156}}$$

By taking the volume integral (d^3x) we find $\frac{dm}{dt} = 0$, i.e. mass conservation.

5.2.3.1 Derivation of the continuity equation*

From

$$\underbrace{\int m\partial_t f d^3u}_{\partial_t\rho(\underline{x},t)} + \underbrace{\underline{\nabla}_{\underline{x}}\int m\underline{u} f d^3u}_{\substack{\text{using } r,u \text{ indep.;}\\ =\underline{\nabla}_{\underline{x}}(\rho\underline{v})}} + \underbrace{\int m\underline{\nabla}_{\underline{u}}(\underline{g}f)d^3u}_{\substack{\text{assume acc. } \underline{g}\\ \text{indep. } \underline{u}}} = \underbrace{\int m\left.\frac{df}{dt}\right|_c d^3u}_{\substack{=0 \text{ (local}\\ \text{particle}\\ \text{conserv.)}}} \tag{157}$$

where the third term also vanishes (apply Gauss, assume $f \to 0$ (e.g. Maxwellian $\exp(-u^2)$) for $u \to \infty$), we get the result from above.

5.2.4 Momentum conservation | Navier Stokes equation (2nd moment)

By following the steps

- multiply Boltzmann equation with momentum ($m\underline{u}$) and integrate over d^3u
- using that collisions conserve momentum
- using the continuity equation

one obtains the Navier-Stokes equation

$$\partial_t(\rho\underline{v}) + \underline{\nabla}\cdot(\rho\underline{v}\underline{v}^T + P\underline{\underline{1}} - \underline{\underline{\Pi}}) = \rho\underline{g} \tag{158}$$

which using the continuity equation becomes

$$\partial_t\underline{v} + (\underline{v}\cdot\underline{\nabla})\underline{v} = \underline{g} - \frac{1}{\rho}\underline{\nabla}P + \frac{1}{\rho}\underline{\nabla}\cdot\underline{\underline{\Pi}} \tag{159}$$

with

$$\begin{gathered}\text{(isotropic) pressure } P = \frac{1}{3}\rho\langle||\underline{w}||^2\rangle\\ \text{anisotropic viscous stress tensor } \Pi_{ij} = P\delta_{ij} - p\langle w_i w_j\rangle \text{ (symmetric and traceless)}\\ \text{acceleration } \underline{g} \text{ from external sources}\end{gathered} \tag{160}$$

Viscosity opposes shearing motion and interprenetation (diffusion of momentum on shear).

Taking the volume integral over the Navier-Stokes equation and applying Gauss theorem, we see that in the absence of an external force ($\underline{g} = 0$), the momentum is conserved.

5.2.4.1 Notes on the derivation of the Navier-Stokes equation

Take a look at the terms in

$$\partial_t \int m\underline{u} f d^3u + \underline{\nabla}_{\underline{x}} \int m\underline{u}\underline{u}^T f d^3u + \int m\underline{g}\underline{u}^T \underline{\nabla}_{\underline{u}} f du^3 = \int m\underline{u} \left.\frac{\partial f}{\partial t}\right|_c d^3u \tag{161}$$

- first term: $\partial_t \int m\underline{u} f d^3u = \partial_t(\rho\underline{v})$ by definition of $\underline{v}$
- second term: $\underline{\nabla}_{\underline{x}} \int m\underline{u}\underline{u}^T f d^3u = \underline{\nabla}_{\underline{x}}\left(\rho\langle\underline{u}\underline{u}^T\rangle\right)$ using $\underline{u} = \underline{v}+\underline{w}$ this becomes $\underline{\nabla}_{\underline{x}}\left(\rho\underline{v}\underline{v}^T\right)+ 2\underline{\nabla}_{\underline{x}}\left(\rho\underline{v}\langle\underline{w}\rangle^T\right) + \underline{\nabla}_{\underline{x}}\left(\rho\langle\underline{w}\underline{w}^T\rangle\right) = \underline{\nabla}_{\underline{x}}\left(\rho\underline{v}\underline{v}^T\right) + \underline{\nabla}_{\underline{x}}\left(\rho\langle\underline{w}\underline{w}^T\rangle\right)$ as $\langle\underline{w}\rangle = \underline{0}$.
- third term: assuming $\underline{g}$ does not depend on $\underline{u}$, $\underline{g}\int m\underline{u}^T\underline{\nabla}_{\underline{u}} f du^3 = -\rho\underline{g}$ as by the chain rule $\underline{u}^T\underline{\nabla}_{\underline{u}} f = \underline{\nabla}_{\underline{u}}(\underline{u}^T f) - f\underline{\nabla}_{\underline{u}}\underline{u}^T$ where the integral over the first term vanishes by the same reasoning as before
- the right-hand side vanishes, because collisions conserve momentum

Finally, $\left(\rho\langle\underline{w}\underline{w}^T\rangle\right)$ (so the correlation matrix of the random fluctuations (?)) is split into an isotropic contribution from pressure P and an anisotropic viscous stress tensor $\underline{\underline{\Pi}}$. The continuity equation can be used to get (first apply chain rule) $\partial_t(\rho\underline{v}) + \underline{\nabla}_{\underline{x}}\left(\rho\underline{v}\underline{v}^T\right) = \rho\left(\partial_t\underline{v} + (\underline{v}\cdot\nabla)\underline{v}\right)$

5.2.4.2 Interpretation and viscous stress tensor for a Newtonian fluid

Our result

$$\frac{d\underline{v}}{dt} = \partial_t\underline{v} + (\underline{v}\cdot\underline{\nabla})\underline{v} = \underline{g} - \frac{1}{\rho}\underline{\nabla}P + \frac{1}{\rho}\underline{\nabla}\cdot\underline{\underline{\Pi}} \tag{162}$$

already looks similar to our intuitive result. We have

- some acceleration from external forces $\underline{g}$ (e.g. gravity)
- pressure force, where on the pressure we now also have a microscopic unterstanding $P = \frac{1}{3}\rho\langle||\underline{w}||^2\rangle$
- some viscous force as described by $\frac{1}{\rho}\underline{\nabla}\cdot\underline{\underline{\Pi}}$

Intuition for viscosity and momentum diffusion: Viscosity is an every-day phenomenon - but how can we understand it microscopically? Consider a fluid with bulk velocity $\underline{v}$ only in x-direction. Now imagine v_x increases with z (maybe because im pulling a plate on top of my container). The molecules constantly bump into each other, and it is much more probabable that a particle from higher up z will give x- momentum to one from lower down in a collision than vice versa. Therefore x-momentum diffuses down. However if the x-velocity increases linearly with z, $v_x(z)$ will not change, as there is as much momentum transport in as out of a given slice along x at some height z. Therefore the simplest diffusion equation for momentum is $\rho\partial_t v_x(z) = \nu\rho\partial_z^2 v_x$ (for a positive curvature more momentum diffuses down from the top then goes out of the bottom). Speaking in terms of the fluctuations $\underline{w}$ from the bulk velocity $\underline{v}$, in the above scenario we expect particles with high *random* downward velocity $-w_z$ to be faster than their surrounding (high w_x), so we expect $-\langle w_z w_x\rangle > 0$.

With the above intuition, it should make sense that

$$\Pi_{ij} = P\delta_{ij} - p\langle w_i w_j\rangle \tag{163}$$

and that we might make the ansatz of Π_{ij} being linear in $\frac{\partial v_i}{\partial x_j}$

$$\begin{aligned}\Pi_{ij} &= \eta D_{ij} + \xi\delta_{ij}(\underline{\nabla}\cdot\underline{v}) \\ D_{ij} &= \frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3}\delta_{ij}(\underline{\nabla}\cdot\underline{v})\end{aligned} \tag{164}$$

with D_{ij} being the deformation tensor that vanishes for uniform expansion or contraction, η and ξ are the coefficients of shear and bulk viscosity respectively, with units $g\,\mathrm{cm}^{-1}\,\mathrm{s}^{-1}$.

- ηD_{ij} represents the resistance to shearing motion
- $\xi\delta_{ij}\left(\underline{\nabla}\cdot\underline{v}\right)$ represents the resistance in volume, remember that for $\underline{\nabla}\cdot\underline{v} = 0$ we have an incompressible fluid

5.2.5 Energy conservation (3rd moment)

By the following steps

- multiply the Boltzmann equation by $m\underline{u}^2$ and integrate over d^3u
- use that collisions conserve energy
- use the continuity equation

one follows

$$\rho\frac{de_{th}}{dt} \equiv \partial_t(\rho e_{th}) + \underline{\nabla}\cdot(\rho e_{th}\cdot\underline{v}) = -P\underline{\nabla}\cdot\underline{v} + \Psi - \underline{\nabla}\cdot\underline{Q}$$
$$\text{specific internal energy } e_{th} \equiv \frac{1}{2}\langle||\underline{w}||^2\rangle$$
$$\text{viscous dissipation rate (bulkmotion } \rightarrow \text{ internal energy) } \Psi \equiv \sum_{i,j=1}^{N}\Pi_{ij}\frac{\partial v_i}{\partial x_j} \tag{165}$$
$$\text{conductive heat flux } \underline{Q} = \frac{1}{2}\rho\langle\underline{w}||\underline{w}||^2\rangle$$

5.2.5.1 Notes on the conductive heat flux

There is only conductive heat flux $\underline{Q}$ is $\underline{w}$ is asymmetric around the bulk motion (hotter (*faster* in the sense of $||\underline{w}||$) particles drift relative to cold ones). The conductive flux is in most cases produced by a temperature gradient

$$\underline{Q} = -\xi\underline{\nabla}T, \quad \text{with } \chi \simeq 6\times10^{-7}T^{\frac{5}{2}}\mathrm{erg\,s^{-1}\,cm^{-1}\,K^{-1}}$$
$$\text{proper diffusion coefficient } \kappa = \frac{\chi T}{P} \text{ in } \mathrm{cm^2\,s^{-1}} \tag{166}$$

(by collisions and random movement, wiggling spreads, stochastically along the gradient of T). As always, there is only change in a volume if there is different in- and outflux, so if $\underline{\nabla}\cdot\underline{Q}\neq 0$.

5.2.5.2 Evolution equation for the total specific energy $e = e_{th} + \underline{v}^2/2$

We are interested in

$$\frac{de}{dt} = \frac{de_{th}}{dt} + \frac{d}{dt}\left(\frac{1}{2}\underline{v}^2\right) \tag{167}$$

where we already know $\frac{de_{th}}{dt}$. So let us consider

$$\begin{aligned}\partial_t\left(\frac{\rho\underline{v}^2}{2}\right) &= \frac{\underline{v}^2}{2}\partial_t\rho + \rho\underline{v}\partial_t\underline{v}\\ &= \frac{\underline{v}^2}{2}\left(-\underline{\nabla}(\rho\underline{v})\right) + \rho\underline{v}\left(-(\underline{v}\cdot\underline{\nabla})\underline{v} + \underline{g} - \frac{1}{\rho}\underline{\nabla}P + \frac{1}{\rho}\underline{\nabla}\cdot\underline{\underline{\Pi}}\right)\end{aligned} \tag{168}$$

where we used the continuity equation and Navier-Stokes equation. Based on

$$(\underline{v}\cdot\underline{\nabla})\underline{v} \equiv \frac{1}{2}\underline{\nabla}\underline{v}^2, \quad \rho\underline{v}\frac{1}{2}\underline{\nabla}\underline{v}^2 + \frac{\underline{v}^2}{2}\underline{\nabla}\rho\underline{v} = \underline{\nabla}\cdot\left(\frac{1}{2}\rho\underline{v}^2\underline{v}\right) \tag{169}$$

this becomes

$$\frac{d}{dt}\left(\frac{\rho \underline{v}^2}{2}\right) = \partial_t\left(\frac{\rho \underline{v}^2}{2}\right) + \underline{v}\cdot\left(\frac{1}{2}\rho\underline{v}^2\underline{v}\right) = \rho\underline{v}\cdot\underline{g} - \underline{v}\cdot\underline{\nabla}P + \underline{v}\cdot\left(\underline{\nabla}\cdot\underline{\underline{\Pi}}\right) \tag{170}$$

so we finally get the evolution equation for the total specific energy $e = e_{th} + \underline{v}^2/2$

$$\partial_t(\rho e) + \underline{\nabla}\cdot\left[(\rho e + P)\underline{v} - \underline{\underline{\Pi}}\cdot\underline{v} + \underline{Q}\right] = \rho\underline{v}\cdot\underline{g} \tag{171}$$

where taking the volume integral and applying Gauss theorem shows energy conservation if $\underline{g} = \underline{0}$.

5.2.6 Entropy conservation

Note: Heat conduction and viscous friction change the entropy and entropy is conserved is those processses are absent.

From the first law of thermodynamics in its specific form (specific volume $\tilde{V} = 1/\rho$)

$$de_{th} = dw + dq = -Pd\tilde{V} + Tds = \frac{P}{\rho^2}d\rho + Tds \tag{172}$$

we find (*divide* by dt and insert the continuity and energy equation)

$$\rho T\frac{ds}{dt} = -\underline{\nabla}\cdot\underline{Q} + \Psi \tag{173}$$

proving the statement in the note.

5.3 Euler Equation and Navier-Stokes equation

Ideal gas dynamics, where we assume internal friction and heat conduction to be absent, are described by the Euler equations. Those assumptions are well justified for gas flow in astrophysics, which are often of extremely low density.

If viscosity is relevant, we use the hydrodynamical equations inclusind viscosity, the Navier-Stokes equation.

We will later discuss fluid instabilities in the absence of viscosity.

5.3.1 Euler Equations

Assuming (valid for non-dense media like air)

- no thermal conductivity
- no internal friction
- no external forces

the balance equations obtained from the moments of the Boltzmann equation simplify to the Euler equations (table 7).

Continuity equation	Balance of momentum	Balance of energy
$\partial_t \rho + \underline{\nabla} \cdot (\rho \underline{v}) = 0 \quad (174)$	$\partial_t(\rho \underline{v}) + \underline{\nabla} \cdot (\rho \underline{v}\,\underline{v}^T + P \underline{\underline{1}}) = 0 \quad (175)$	$\partial_t(\rho e) + \underline{\nabla} \cdot \left[(\rho e + P)\underline{v}\right] = 0$ with $e = e_{th} + \frac{1}{2}\underline{v}^2$ $\frac{\text{total energy}}{\text{unit mass}} \quad (176)$

Table 7: Euler equations

The equations form a set of hyperbolic conservation laws (all continuity equations, one for mass, one for momentum, one for energy).

Hierarchy and closure The equations are in a hierarchy that in one equations occur quantities following from the next higher one (velocity in the continuity equation, ...) - we need a further closure relation making the energy balance unnecessary, for an ideal gas this is

$$P = (\gamma - 1)\rho e_{th}, \quad \gamma = \text{ specific heat ratio } = \frac{c_p}{c_v}, \quad \text{for monoatomic gas } \gamma = \frac{f+2}{f} = \frac{5}{3} \tag{177}$$

5.3.2 Navier-Stokes equation

In real fluids, viscosity transforms relative motion into heat. For

- vanishing conductivity
- vanishing external forces

we have (table 8)

$\underline{\underline{\Pi}}$ is the viscous stress tensor, a material property. For a vanishing stress tensor, we recover the Euler equations. To first order $\underline{\underline{\Pi}}$ must be linear in the velocity derivatives, where the

Continuity equation	Balance of momentum	Balance of energy
$\partial_t \rho + \underline{\nabla} \cdot (\rho \underline{v}) = 0$ (178)	$\partial_t(\rho \underline{v}) + \underline{\nabla} \cdot (\rho \underline{v}\underline{v}^T + P\underline{\underline{1}}) = \underline{\nabla} \cdot \underline{\underline{\Pi}}$ (179)	$\partial_t(\rho e) + \underline{\nabla} \cdot \left[(\rho e + P)\underline{v}\right] = \underline{\nabla} \cdot \left(\underline{\underline{\Pi}} \cdot \underline{v}\right)$ (180)

Table 8: Navier stokes equations

most general rank-2 tensor of this type can be written as

$$\begin{gathered}\underline{\underline{\Pi}} = \eta \left[\underline{\nabla}\underline{v}^T + (\underline{\nabla}\underline{v}^T)^T - \frac{2}{3}(\underline{\nabla} \cdot \underline{v})\underline{\underline{1}}\right] + \xi(\underline{\nabla} \cdot \underline{v})\underline{\underline{1}} \\ \eta \text{ scales the traceless part, describes shear viscosity} \\ \xi \text{ scales the trace, describes bulk viscosity} \\ \text{possibly } \eta, \xi \text{ depend on } \rho, T, \dots\end{gathered} \tag{181}$$

5.3.2.1 Simplification of the Navier-Stokes equations for incompressible fluids ($\underline{\nabla} \cdot \underline{v} = 0$)

In the case $\underline{\nabla} \cdot \underline{v} = 0$ only shear forces matter (no bulk compression), and we have

$$\frac{1}{\eta}(\underline{\nabla} \cdot \underline{\underline{\Pi}})_x = \left(\underline{\nabla} \cdot \left[\underline{\nabla}\underline{v}^T + (\underline{\nabla}\underline{v}^T)^T\right]\right)_x = \underline{\nabla}^2 v_x \tag{182}$$

(hint: write the component out and use $\partial_x^2 v_x + \partial_y \partial_x v_y + \partial_z \partial_x v_z = \partial_x(\partial_x v_x + \partial_y v_y + \partial_z v_z) = 0$). Introducing the kinematic viscosity $\nu \equiv \frac{\eta}{\rho}$, we get

$$\frac{d\underline{v}}{dt} = \partial_t \underline{v} + (\underline{v} \cdot \underline{\nabla})\underline{v} = -\frac{\underline{\nabla} P}{\rho} + \nu \underline{\nabla}^2 \underline{v} \tag{183}$$

so the simplified expression from the beginning. The bulk motion responds to pressure gradients and viscous forces.

5.3.2.2 Characterizing flow | Reynolds number

Consider a flow problem with characteristic length scale L_0, velocity V_0 and density scale ρ_0. Let us define dimensionless fluid variables and operators

$$\begin{aligned} \hat{\underline{v}} = \frac{\underline{v}}{V_0}, \quad \hat{\underline{x}} = \frac{\underline{x}}{L_0}, \quad \hat{P} = \frac{P}{\rho_0 V_0^2} \\ T_0 = \frac{L_0}{V_0}, \quad \hat{t} = \frac{t}{T_0}, \quad \hat{\rho} = \frac{\rho}{\rho_0}, \quad \hat{\underline{\nabla}} = L_0 \underline{\nabla} \end{aligned} \tag{184}$$

where for the pressure mind that pressure is also an energy density and the kinetic energy density is $\frac{1}{2}\rho_0 V_0^2$.

Plugging this the incompressible Navier-Stokes equation (183) we get

$$\frac{d\hat{\underline{v}}}{d\hat{t}} = -\frac{\hat{\underline{\nabla}} P}{\hat{\rho}} + \frac{\nu}{L_0 V_0} \hat{\underline{\nabla}}^2 \hat{\underline{v}} \tag{185}$$

involving the Reynolds number

$$Re \equiv \frac{L_0 V_0}{\nu} \tag{186}$$

We can understand the Reynolds number as the ratio between the chaotic advective term in the Navier-Stokes equation and the frictional term, so

$$Re = \frac{\text{advective term}}{\text{frictional term}} = \frac{|(\underline{v} \cdot \underline{\nabla})\underline{v}|}{\nu \underline{\nabla}^2 \underline{v}} \tag{187}$$

Note: Connection between the Reynolds number and turbulence: The quadratic (in the velocity) term $(\underline{v} \cdot \underline{\nabla})\underline{v}$ generates turbulence (deterministic chaos) while the viscous term $\nu \underline{\nabla}^2 \underline{v}$ destroys it via dissipation. In terms of energy

$$Re \sim \frac{V_0 L_0}{\nu} = \frac{\rho L^3 U^2}{\mu L^2 U} = \frac{2E_{kin}}{W_{friction}}, \quad \text{as } W_{friction} \sim F_{friction} \cdot L_0 = V \rho \nu \underline{\nabla}^2 \underline{v} \cdot L_0 = \mu V_0 L_0^2 \tag{188}$$

(with the kinematic viscosity $\nu = \frac{\mu}{\rho}$ in $m^2\,\mathrm{s}^{-1}$). In general, the higher the Reynolds number the more turbulence, for

$$Re > R_c \sim 10^3 \tag{189}$$

we have turbulent flow (figure 40). For $Re \sim 1$ viscosity dominates, for $Re \to \infty$ we approach an ideal gas.

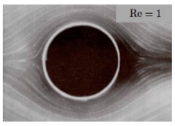
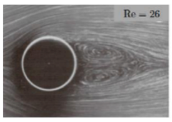
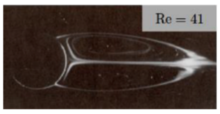
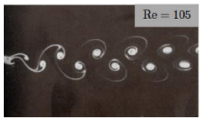
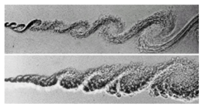
Re = 1 (domination of viscous friction)	Re = 26	Re = 41	Re = 105 (still regular)	Re > 10^3 Turbulence (higher Re → turbulence on smaller scales)
				

Figure 40: Reynolds number

5.4 Shocks

In hydrodynamical flows, shock waves can develop, where the fluid variables

- density ρ
- velocity $\underline{v}$
- temperature T
- specific entropy s

jump by finite amounts. In the frame of the Euler equations these are true mathematical discontinuities, while exhibiting a finite width in the Navier Stokes equations. A shock wave

- propagates faster than the signal speed for compressible waves c_s
- produces and irreversible change to the fluid change (increase in entropy)

A shock wave is a region of small thickness over which the properties of the flow change rapidly.

5.4.1 Propagation of disturbances 1: Speed of sound

Microscopic Intuition: Consider a higher density region in a fluid. Probabalisticly there is more momentum in the direction of lower density, so (by collisions) the higher density will spread out. The characteristic speed of on which density information propagates is related to the jiggling of the particles, i.e. their themperature. This characteristic speed of sound is roughly derived in the following.

Soundwaves are messengers, carrying density and pressure fluctuations. Imagine you're driving in fog towards a traffic jam - if you're so quick no messenger can quickly enough reach you, you will have a shock.

Consider the Euler equation for momentum without internal forces or friction in 1D in a steady state with constant flux $j = \rho v$, so $0 = d(\rho v) \to \rho dv = -v d\rho$. We get

$$\frac{dv}{dt} = -\frac{1}{\rho}\frac{dP}{dx} \quad \to \quad dP = -(\rho dv)\frac{dx}{dt} = (v d\rho)v \quad \to \quad v^2 \equiv c_s^2 = \frac{dP}{d\rho} \tag{190}$$

(note is relative to the bulk motion, so in the local rest frame of the flow, more later).

Now we use that for an adiabatic process, $P\rho^{-\gamma} = \text{const.}$ (and $T\rho^{1-\gamma} = \text{const.}$) so taking the logarithm and differentiating with respect to ρ yields

$$\frac{dP}{d\rho} = \frac{\gamma p}{\rho} \tag{191}$$

so using the ideal gas law

$$P = nk_B T = \frac{\rho}{m_p \mu} k_B T \quad \text{with } \mu \text{ being a mean molecular weight} \tag{192}$$

we get

$$c_s^2 = \frac{\gamma k_B T}{m_p \mu} \tag{193}$$

so based on $T\rho^{1-\gamma} = \text{const.}$ we can write

$$c_s^2 \propto \rho^{\gamma-1} \tag{194}$$

5.4.2 Characteristics of Perturbations

Idea: Our aim is to find the characteristics, lines in the (x, t) plane along which perturbations propagate.

Let us start with the continuity equation in 1D

$$\text{cont.:} \ \frac{1}{\rho}\left(\frac{\partial \rho}{\partial t} + v\frac{\partial \rho}{\partial x}\right) = 0 \tag{195}$$

With $c_s^2 = \frac{\partial P}{\partial \rho}$ we can write the Euler equation for momentum as

$$\text{Euler:} \ \frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} = -\frac{1}{\rho}\frac{\partial P}{\partial x} = -\frac{c_s^2}{\rho}\frac{\partial \rho}{\partial x} \tag{196}$$

Based on $c_s^2 \propto \rho^{\gamma-1}$, we can replace $\partial\rho$ in those equations using

$$\frac{d\rho}{\rho} = \frac{2}{\gamma - 1}\frac{dc_s}{c_s} \tag{197}$$

With this replace $\partial\rho$ in the continuity and Euler equation. From adding and subtracting the Euler and continuity equation, we get

$$\begin{aligned} \left[\partial_t + (v + c_s)\,\partial_x\right]\left(u + \frac{2}{\gamma-1}c_s\right) &= 0 \\ \left[\partial_t + (v - c_s)\,\partial_x\right]\left(u - \frac{2}{\gamma-1}c_s\right) &= 0 \end{aligned} \tag{198}$$

Defining

$$\begin{aligned} \xi_+ \equiv u + \frac{2}{\gamma-1}c_s \quad &\rightarrow \quad \frac{d}{dt}\xi_+(x(t),t) = \left[\partial_t + (\partial_t x)\partial_x\right]\xi_+(x(t),t) = 0 \text{ for } \partial_t x = v + c_s \\ \xi_- \equiv u - \frac{2}{\gamma-1}c_s \quad &\rightarrow \quad \frac{d}{dt}\xi_-(x(t),t) = \left[\partial_t + (\partial_t x)\partial_x\right]\xi_-(x(t),t) = 0 \text{ for } \partial_t x = v - c_s \end{aligned} \tag{199}$$

where we applied the *method of characteristics*.

> From this we can see that (*as expected*) in a fluid with bulk motion v, perturbations propagate along characteristics with velocity $v \pm c_s$.

These characeteristic equations are the same no matter the aplitude of the perturbations.

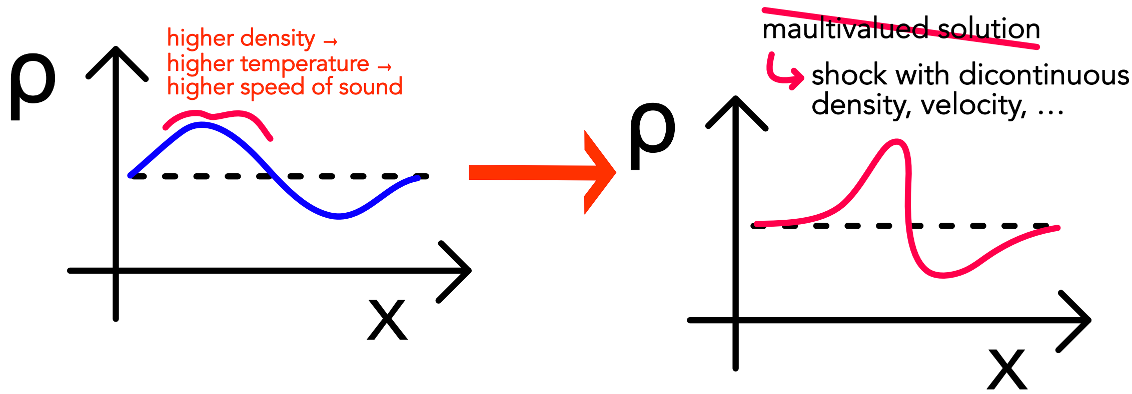


Figure 41: Formation of a shock

5.4.3 Formation of a shock

5.4.3.1 Formation as a pressure driven compressive disturbance

Consider a fluid with base density ρ_0. For small perturbation in density, we can to first order use

$$c_s = \left(\frac{\partial P}{\partial \rho}\right)^{\frac{1}{2}} \simeq \frac{\gamma p_0}{\rho_0} \tag{200}$$

But now consider a larger perturbation. Adiabatically the temperature scales with $T \propto \rho^{\gamma-1}$, and as c_s^2 scales linearly with the temperature, we have

$$c_s^2 \propto T \propto \rho^{\gamma-1} \tag{201}$$

Therefore, the "waves crest overtakes the valley" (figure 41) but as the hydrodynamic equations don't allow for multivalued solutions, we get a shock with discontinuities in ρ, v, T and s. For an isothermal gas $c_s = \text{const.}$ but steepening can happen nonetheless as of the non-linearity of the Euler / Navier-Stokes equations (?).

5.4.3.2 Causes for shocks

- supersonic compressible disturbance
- supersonic collision of two streams of fluids
- non-linear interaction of subsonic compressible modes (nonliear wave interaction)

5.4.4 Collisional and collisionless shocks | shock front

The »discontinuous« change normally happens over a scale proportional to the *effective* mean free path λ_{eff}.

- Collisional shocks: Coulomb-collisions determine λ_{eff}
- Collisionless plasma like solar wind: λ_{eff} is reduced by electromagnetic viscosities $\lambda_{eff} \ll \lambda_{coulomb}$

The shock front or transition layer is of the scale of λ_{eff}. Here, viscous effects are important - they dissipate kinetic energy, generating heat and entropy. Outside this layer viscous effects are small on scales $L \gg \lambda_{eff} \rightarrow \underline{\underline{\Pi}} = \underline{\underline{0}}$.

Note: This scale violates the assumptions under which the Navier-Stokes equations are derived from the Boltzmann equation. »It would be more than 50 years before computer simulation and laboratory experiments would show that physical shocks are measured to be twice the width predicted by theory, validating Becker's assertion that something beyond the Navier-Stokes description is needed.« (Margolin and Lloyd-Ronning, 2023)

5.4.5 Properties at fluid discontinuities

External gravitational forces and conductive heat flux are way slower than the transition time of fluid discontinuities and can thus be neglected.

We consider the propagating fluid discontinuity in its rest frame (i.e. upstream is ahead of the shock).

We distinguish two types of fluid discontinuities

- shocks characterized by mass flux through their interface
- contact dicontinuities without such mass flux

5.4.5.1 (Rankine-Hugoniot) Jump conditions I: Assumptions

Relative to the shock, fluid moves from upstream to downstream and we would like to relate upstream conditions ρ_1, v_1, T_1 to downstread conditions ρ_2, v_2, T_2 by *jump conditions*, see figure 42.

We assume

- the velocity to be perpendicular to the surface of the discontinuity (we later generalize)
- a steady state $\partial_t = 0$
- a 1D situation $\underline{\nabla} \rightarrow \partial_x$

5.4.5.2 (Rankine-Hugoniot) Jump conditions II: Jump condition from the continuity equation

From the above assumption and the continuity equation, we following

$$\begin{gathered} \frac{d}{dx}(\rho v) = 0 \rightarrow \rho v = \text{const.} \rightarrow \rho_1 v_1 = \rho_2 v_2 = j, \quad \text{mass flux } j \\ v_1, v_2 \text{ measured in frame of the discontinuity} \\ \text{notation for up-downstream-difference: } \boxed{[\rho v] = 0} \end{gathered} \tag{202}$$

The mass flux is constant across the discontinuity.

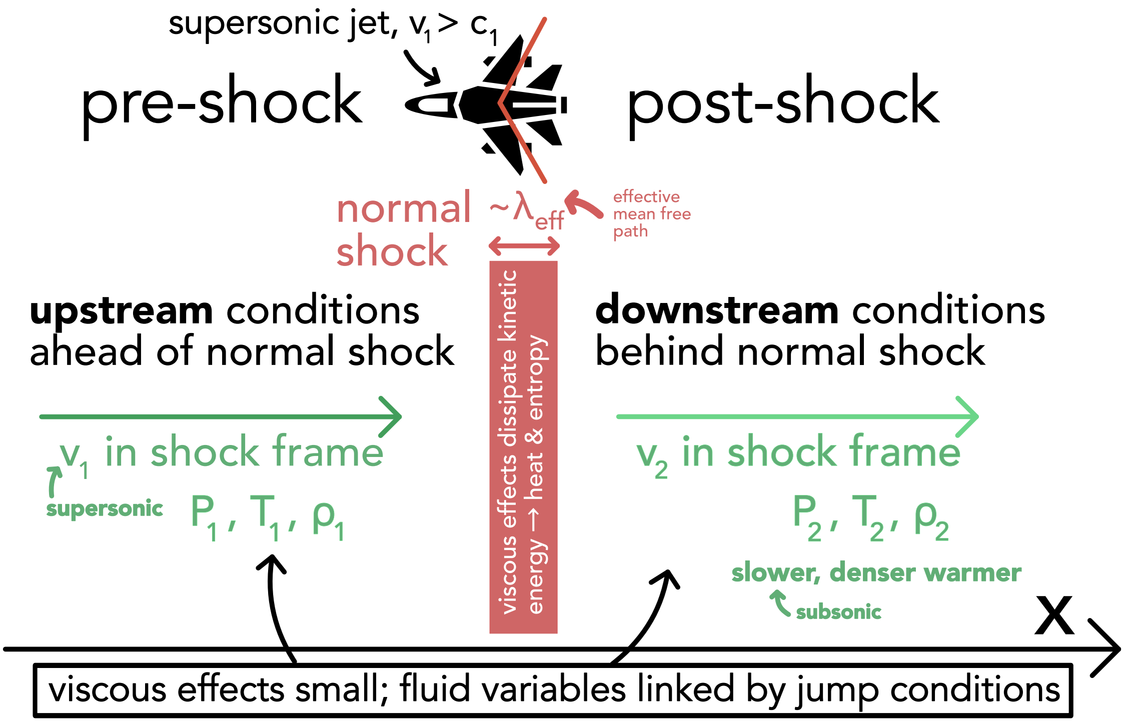


Figure 42: Normal shock

5.4.5.3 (Rankine-Hugoniot) Jump conditions III: Jump condition from the momentum equation

One obtains (for the pre and post shock zones)

$$\frac{d}{dx}(\rho v^2 + P) = 0 \quad \rightarrow \quad \boxed{[\rho v^2 + P] = 0} \tag{203}$$

Note: We consider the difference in the pre- and post-shock-zones, where viscosity effects can be neglected ($\xi, \eta = 0$) and also $\frac{dv}{dx} = 0$. Note that as the transition zone is typically of a scale λ_{mfp}, we would have to resort to kinetic theory (or plasma particle in cell-codes) there anyways.

5.4.5.4 (Rankine-Hugoniot) Jump conditions IV: Jump condition from the energy equation

We obtain

$$
\begin{aligned}
0 &= \frac{d}{dx}\left((\rho e + P)v\right) = \frac{d}{dx}\left(\rho v\left(e_{th} + \frac{v^2}{2} + \frac{P}{\rho}\right)\right) \\
&= \rho v \frac{d}{dx}\left(\left(e_{th} + \frac{v^2}{2} + \frac{P}{\rho}\right)\right) + \frac{d(\rho v)}{dx}\left(\left(e_{th} + \frac{v^2}{2} + \frac{P}{\rho}\right)\right) \underbrace{=}_{[\rho v]=0} \rho v \frac{d}{dx}\left(\left(e_{th} + \frac{v^2}{2} + \frac{P}{\rho}\right)\right) \\
&\rightarrow \boxed{\left[\frac{v^2}{2} + e_{th} + \frac{P}{\rho}\right] = 0}
\end{aligned}
\tag{204}
$$

Marked in the $\boxed{\text{boxes}}$ are the Rankine-Hugoniot Jump conditions.

We can plug in the closure $e_{th,i} = P_i / (\rho_i \cdot (\gamma_i - 1))$ into the energy jump condition where theoretically γ_i can be different in the pre- and post-shock zones, e.g. when molecules are dissociated.

5.4.5.5 Types of dicontinuities: contact discontinuity vs. shock

The continuity of mass flux allows two scenarios

- **tangential discontinuity**: $\rho_1 v_1 = \rho_1 v_2 = 0$ so as $\rho_1, \rho_2 \neq 0$ in general, $v_1 = v_2 = 0$ so $[P] = 0$ as of $[\rho v^2 + P] = 0$. The pre- and post-shock zones move with the same velocity as the shock, there is no mass-flux through the shock. If the tangential velocities are also continuous ($[v_y] = [v_z] = 0$) (which we do not consider by our current assumptions), this is called a contact dicontinuity. The density may jump but as of $[P] = 0$, T then has to do the *opposite* jump. A contact discontinuity is a surface separating two fluids with different physical properties.
- **shock**: for $\rho_1 v_1 = \rho_1 v_2 \neq 0$ we have mass flux and thus a shock. Shock waves do propagate with respect to the fluid because of the mass flux in the normal.

5.4.6 Characterizing the Shock strength - Mach number

Note: In the shock's frameof reference, the unshocked material is moving at speed v_1 and the shocked material at speed v_2.

The (pre-shock) Mach number is the ratio between the upstream (with respect to the shock) velocity to the upstream sound speed, characterizing the strength of a shock ($\rho_1 v_1 \neq 0$) (in case of a jet causing the upstream velocity, the jets velocity is used[6])

$$\mathcal{M}_1 \equiv \frac{v_1}{c_{s,1}} \underbrace{=}_{c_s^2=\frac{\gamma P}{\rho}} \sqrt{\frac{\rho_1 v_1^2}{\gamma P_1}} \underbrace{=}_{P=\frac{\rho}{m}k_B T} \sqrt{\frac{m v_1^2}{\gamma k_B T_1}} \tag{205}$$

we can analogously define a post-shock Mach number $\mathcal{M}_2$.

$$\mathcal{M}_2 \equiv \frac{v_2}{c_{s,2}} \tag{206}$$

Equivalently, the Mach number is

- ratio of ram pressure ρv^2 (pressure as of fluids bulk motion, not the thermal motion) to thermal pressure (for $\mathcal{M}_1$ in the pre-shock zone)
- and (as pressure is also an energy density) the kinetic thermal energy density

Note: Below $\mathcal{M} = 1$ there is no shock, the flow is subsonic. A shock occurs, when supersonic flow (e.g. Solar Wind) encounters an obstacle forcing a change in velocity (e.g. the Earth's magnetosphere $\rightarrow$ bow shock).

5.4.6.1 Occurence of the Mach number in the continuity equation

Rewriting $\partial_t \rho + \underline{\nabla}\left(\rho \underline{v}\right) = 0$ using $\frac{D}{Dt} = D_t = \partial_t + \underline{v} \cdot \underline{\nabla}$ to $-\frac{1}{\rho}\frac{D\rho}{Dt} = \underline{\nabla} \cdot \underline{v}$ and using the adiabatic $dp = c_s^2 d\rho$ we get

$$-\frac{1}{\rho c_s^2}\frac{D\rho}{dt} = \underline{\nabla} \cdot \underline{v} \xrightarrow{\text{dimensionless form}} -\mathcal{M}^2 \frac{1}{\hat{\rho}}\frac{D\hat{\rho}}{D\hat{t}} = \underline{\hat{\nabla}} \cdot \underline{\hat{v}} \tag{207}$$

So in the limit $\mathcal{M} \rightarrow 0$ we have incompressible flow.

5.4.6.2 Rewriting the Rankine-Hugoniot jump conditions in terms of $\mathcal{M}_1$ - relating pre- and post-shock quantities

One can rewrite the jump conditions in terms of the Mach number $\mathcal{M}_1$ (assume $\gamma_1 = \gamma_2 = \gamma$) (here without proof)

[6] If a jet is flying sufficiently fast, some of its energy goes into compressing the air in front. If the jet itself moves faster than c_s, the information speed in the air, shock waves form as of those compressions (they cannot spread suffieciently fast for there not to be a shock).

$$\begin{aligned}
\frac{\rho_2}{\rho_1} &= \frac{v_1}{v_2} = \frac{(\gamma+1)\mathcal{M}_1^2}{(\gamma-1)\mathcal{M}_1^2+2} \xrightarrow{\gamma=1} \mathcal{M}_1^2 \\
\frac{P_2}{P_1} &= \frac{\rho_2 k_\mathrm{B} T_2}{\rho_1 k_\mathrm{B} T_1} = \frac{2\gamma\mathcal{M}_1^2-(\gamma-1)}{\gamma+1} \xrightarrow{\gamma=1} \mathcal{M}_1^2 \\
\frac{T_2}{T_1} &= \frac{\left[(\gamma-1)\mathcal{M}_1^2+2\right]\left[2\gamma\mathcal{M}_1^2-(\gamma-1)\right]}{(\gamma+1)^2\mathcal{M}_1^2} \xrightarrow{\gamma=1} 1
\end{aligned} \tag{208}$$

from which we for a strong shock ($\mathcal{M}_1 \gg 1$)[7] can find (use $\gamma = 5/3$ for an ideal non-relativistic gas)

$$\begin{aligned}
\frac{\rho_2}{\rho_1} &= \frac{v_1}{v_2} \approx \frac{\gamma+1}{\gamma-1} = 4, \\
P_2 &\approx \frac{2\gamma}{\gamma+1}\mathcal{M}_1^2 P_1 = \frac{2}{\gamma+1}\rho_1 v_1^2 = \frac{3}{4}\rho_1 v_1^2, \\
k_\mathrm{B} T_2 &\approx \frac{2\gamma(\gamma-1)}{(\gamma+1)^2} k_\mathrm{B} T_1 \mathcal{M}_1^2 = \frac{2(\gamma-1)}{(\gamma+1)^2} m v_1^2 = \frac{3}{16} m v_1^2,
\end{aligned} \tag{209}$$

In the shock frame, the post-shock medium is slower, denser, has higher-pressure and is warmer, see figure 43.

5.4.6.3 Conversion of kinetic to thermal energy in the shock

Based on the above relations for $\mathcal{M}_1 \gg 1, \gamma = 5/3$ we can write

$$\begin{aligned}
\text{post-shock specific kinetic energy: } & \frac{1}{2}v_2^2 \approx \frac{1}{16}\frac{1}{2}v_1^2 \\
\text{post-shock specific thermal energy: } & \frac{3}{2}\frac{k_B T_2}{m} \approx \frac{9}{32}v_1^2 = \frac{9}{16}\frac{1}{2}v_1^2
\end{aligned} \tag{210}$$

We find that in the shock frame, roughly half of the pre-shock kinetic energy ($\frac{9}{16}$) is converted to thermal energy.

5.4.6.4 Conservation of energy in the shock

In the pre-shock flow (for a strong shock) we can neglect the thermal energy, so $e_1 = \frac{v_1^2}{2}$ (specific energy per particle).

In

$$\frac{1}{2}v_2^2 + \frac{3}{2}\frac{k_B T_2}{m} \approx \frac{10}{16}\frac{v_1^2}{2} \tag{211}$$

(in the shock rest frame) one is missing the pdV work, which is done by the shock to compress the post-shock gas and amounts to $k_B T_2 \approx \frac{6}{16}\frac{1}{2}v_1^2$.

[7] In a strong shock $v_1^2 \gg c_1^2$, so the thermal pressure of the unshocked gas is negligible to its ram pressure.

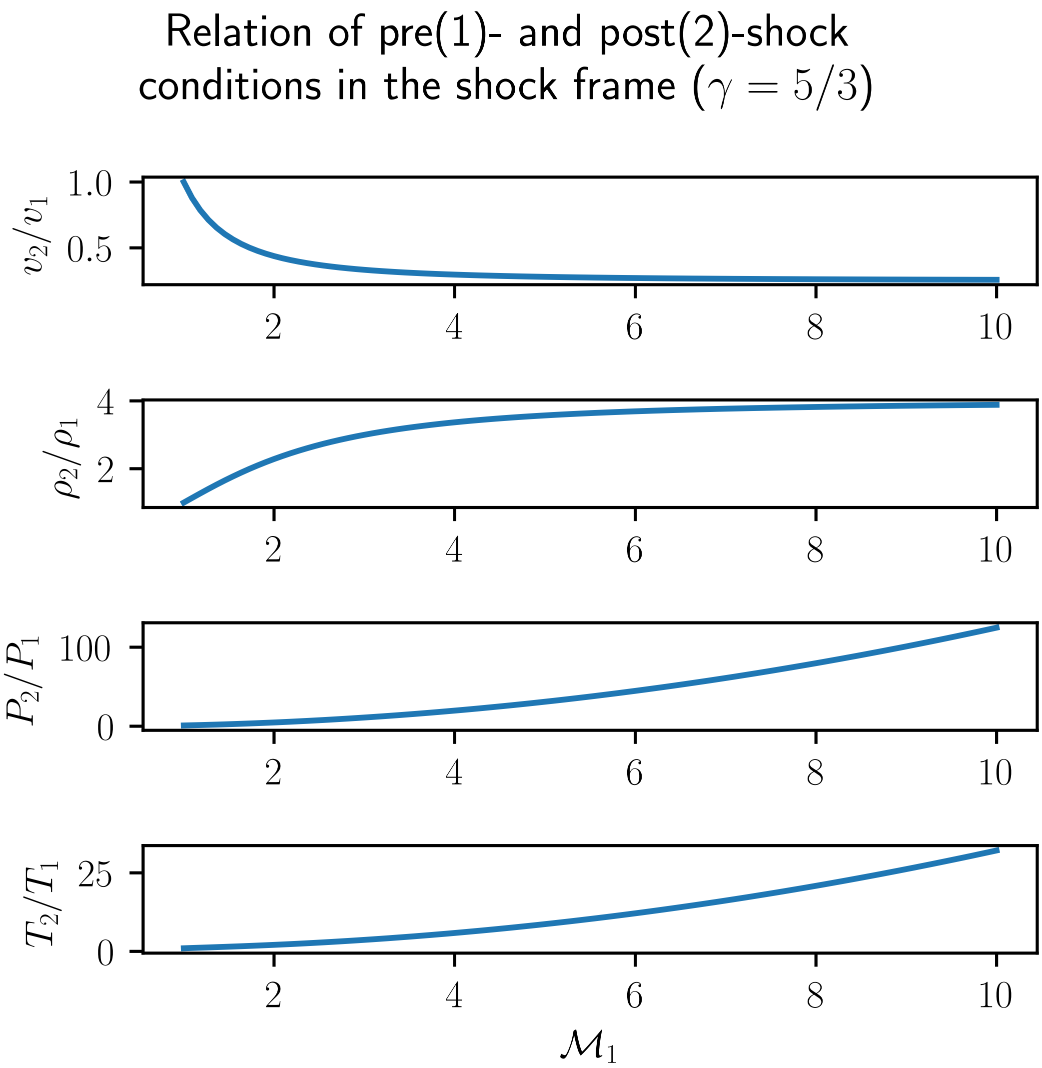


Figure 43: Relation of pre- and post-shock quantities (shock $\rightleftarrows \mathcal{M}_1 = \frac{v_1}{c_1} > 1$). v_1 is the velocity of the upstream (pre-shock) fluid with respect to the shock.

Note: The sum of enthalpy (thermal energy + pdV work) and kinetic energy is conserved in adiabatic flow (even when non-adiabatic processes like shocks occur between the two sections). Enthalpy plays the same role in a flowing system that internal energy takes in a non-flowing one, taking care of the energy associated with flow work in / out of the control volume.

Note: There are also radiative shocks, where in the transition through the shock energy is radiated away.

Note: In the rest frame of the post-shock gas there is no PdV term.

5.4.6.5 Connection between pre- and post-shock Mach number

We can write the post-shock Mach number as

$$\mathcal{M}_2 = \frac{v_2}{c_2} = \frac{v_1}{c_1}\frac{v_2}{v_1}\frac{c_1}{c_2} \underbrace{=}_{c^2 \propto T} \mathcal{M}_1 \frac{v_2}{v_1}\left(\frac{T_1}{T_2}\right)^{\frac{1}{2}} \tag{212}$$

Plugging in the jump condition for T_2/T_1, in the strong shock limit we get

$$\mathcal{M}_2 = \left(\frac{\gamma - 1}{2\gamma}\right)^{\frac{1}{2}} \underbrace{\approx}_{\gamma=5/3} 0.45 \tag{213}$$

Summary: Supersonic gas is slowed down (to subsonic), compressed (density, pressure and temperature increase) by a shock.

5.4.6.6 Shock adiabatic curve*

Note: In shocks, the post-shock entropy is increased with respect to the pre-shock entropy - the shock shifts the gas to a higher adiabatic curve.

The shock is a non-adiabatic process ($\delta Q \neq 0 \rightarrow dS = \frac{\delta Q}{T} \neq 0$). Based on the first law of thermodynamics $\delta Q = dE + PdV$, the ideal gas law and $dE = \nu c_V dT$ (number of mols ν), we can one can find for an ideal polytropic gas that $s = c_V \ln\left(\frac{p}{\rho^\gamma}\right)$, so here

$$s_2 - s_1 = c_V \ln\left(\frac{p_2}{p_1}\left(\frac{\rho_1}{\rho_2}\right)^\gamma\right) = c_V \ln\left(\frac{K_2}{K_1}\right) \tag{214}$$

The shock shifts the gas to a higher adiabatic curce $K = P\rho^{-\gamma}$. Note that one finds using the jump conditions, that $s_2 - s_1 > 0$ only for $\mathcal{M}_1 > 1$, so there is only a shock for $\mathcal{M}_1 > 1$.

Based on $j = \rho_1 v_1 = \rho_2 v_2$ and $[\rho v^2 + P] = 0$ the slope of the shock adiabatic curve in a PV-diagram is

$$\frac{j^2}{m} = \frac{P_2 - P_1}{V_1 - V_2} \tag{215}$$

$P = \text{const.} \times \rho^\gamma$ on either side of the shock (where we assume equilibrium), but with different constants.

Note: The jump conditions are *reversible* - if we interchange post- and pre-shock flow conditions, so $v_1 < v_2$, then density, pressure and temperature would decrease across the shock and so the entropy, excluding a shock with deceleration.

5.4.6.7 Oblique shocks

The fluid might not impact the shock perpendicular to the shock front, but at an oblique angle. **Result:** The shock deflects the flow away from the shocks normal direction (towards the shock's surface), the final velocity may remain supersonic. Only the velocity component normal to the shock front changes, V_t is continuous across the shock (see figure 44).

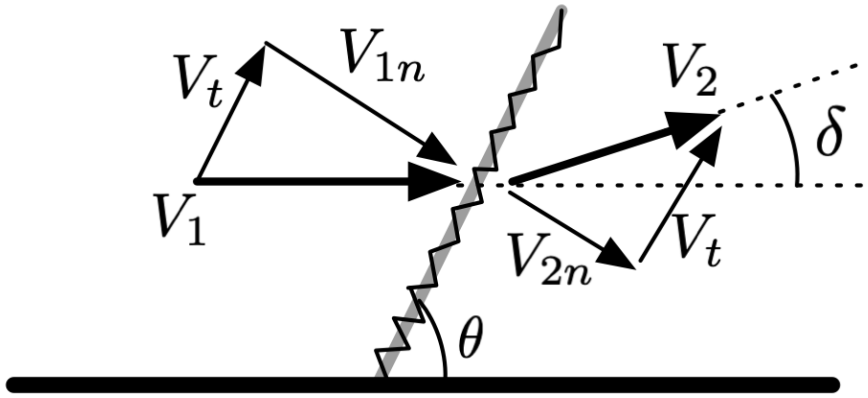


Figure 44: Oblique shock

Derivation - oblique jump conditions: Let $\underline{n}$ (here $= \underline{\hat{e}}_x$) be the shock normal, so $v_\parallel = \underline{v} \cdot \underline{n}$ is the component of the fluid velocity v parallel and $v_\perp$ the one perpendicular to $\underline{n}$. The Navier-Stokes equation describes conservation of momentum in form of a continuity equation and as such contains a momentum current. This current across the shock, $\rho \underline{v}(\underline{v} \cdot \underline{n})$, is continuous across the shock, yielding

$$\begin{aligned} [\rho v_x^2 + P] &= 0 \\ [\rho v_x v_y] &= 0 \\ [\rho v_x v_z] &= 0 \end{aligned} \tag{216}$$

As we are dealing with a shock with momentum flux $[\rho v_x] \neq 0$, we get

$$[v_y] = 0, \quad [v_z] = 0 \tag{217}$$

so as stated, the tangential velocities are continuous across the shock, $v_{1,\perp} = v_{2,\perp} = v_\perp$ and for $v_{2,\parallel}$ we have $v_{2,\parallel} = v_{1,\parallel} \frac{\rho_1}{\rho_2}$ $(\rho_2 > \rho_1)$. So if the component parallel to the shock normal gets smaller and the one perpendicular remains the same, we are deflected away from the shock normal.

5.5 Fluid instabilities

Instabilities are the rapid growth of small perturbations, tapping into a source of free energy.

5.5.1 Stability of a shear flow

We consider two flows counterpropagating side by side (see figure 45). Using perturbation theory, the stability of the flow can be analyzed - if the dispersion relation for a perturbation yields a positive imaginary part (not just yeal oscillation) such modes grow exponentially - the flow is unstable.

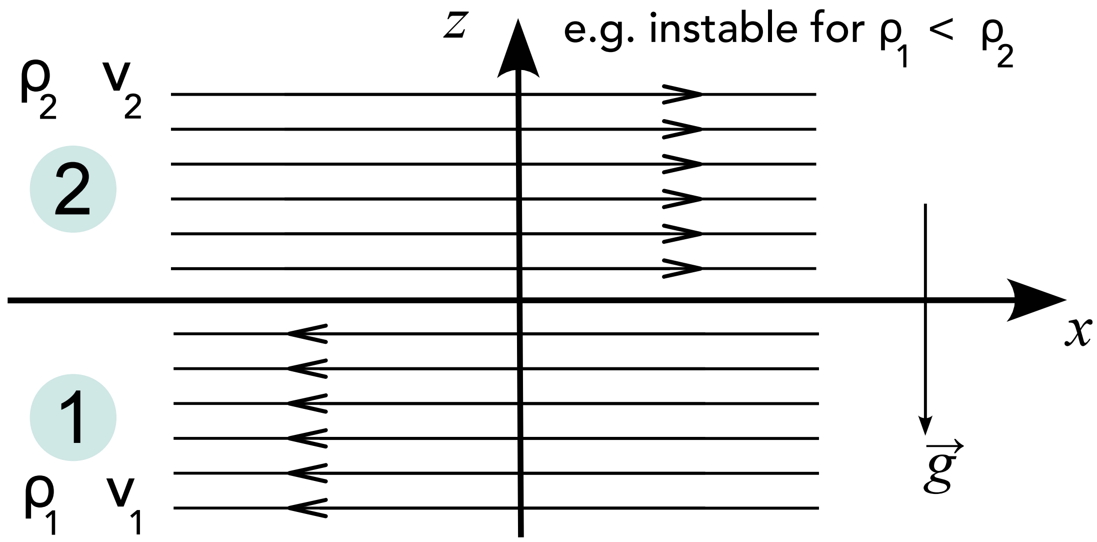


Figure 45: Shear flow

5.5.2 Rayleigh-Taylor instability

Here we consider a fluid at rest, i. e. $v_1 = v_2 = 0$. If the denser fluid lies on top, there are unstable solution - Rayleigh-Taylor instability. The instability is driven by the buoyancy of the lighter fluid or rather the release of potential energy with respect to the external force $\underline{g}$ (see figure 46). If the denser fluid is on the bottom, the interface is stable and will only oscillate if perturbed. There is also the Rayleigh-Taylor instability in plasmas.

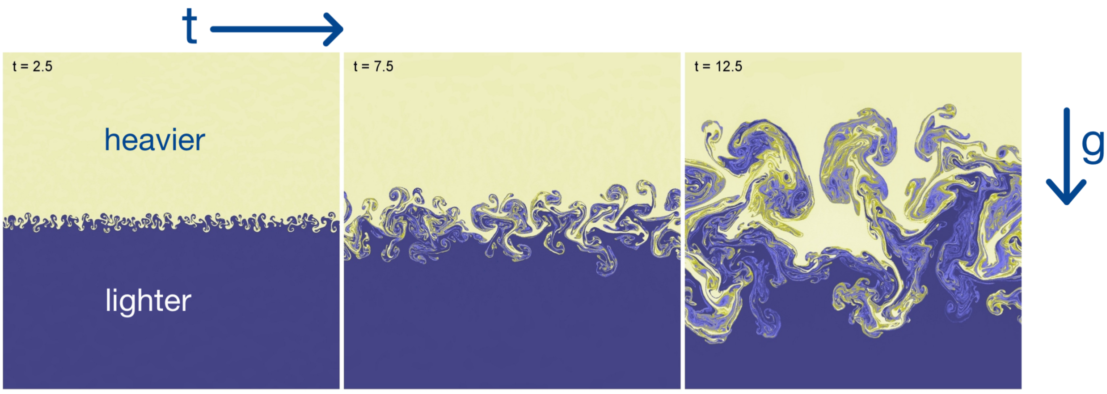


Figure 46: Rayleigh-Taylor instability

5.5.3 Kelvin-Helmholtz instability

Consider the case without the gravitational field $\underline{g} = 0$. In an ideal gas, small wave like perturbances will grow into large waves (with largest growth for large k, so small wavelengths) - Kelvin-Helmholtz-Billows, which subsequently roll up to vortex-like structures. Sharp velocity gradients are unstable - we can create turbulence. Some modes can be stabilized against the instability if we have the gravitational field, the heavy part is on the bottom (otherwise Rayleigh-Taylor instability) and the velocity difference $(v_2 - v_1)^2$ is sufficiently small.

5.5.4 Further instabilities

- **Richtmyer-Meshkov instability**: at suddenly accelerated interfaces
- **Jeans-instability**: in self-gravitating fluids, where denser regions can grow and collapse under their own attraction
- **Thermal instbaility**: ...

5.6 Turbulence

»Big whirls have little whirls that feed on their velocity, and little whirls have lesser whirls and so on to viscosity.« - Lewis Fry Richardson

Both laminar and turbulent flow are solutions to the deterministic Navier-Stokes equations, however turbulent flow is chaotic, laminar not (see table 9 and figure 47).

Laminar flow	**Turbulent flow**
Fluid flows in parallel layers with no disruption between those layers	Unsteady, chaotic flow with varying velocity and pressure in position and time
similar conditions $\rightarrow$ similar solutions	infinitesimal difference in conditions $\rightarrow$ vastly different solutions (deterministic chaos[8])

Table 9: Laminar vs. turbulent flow

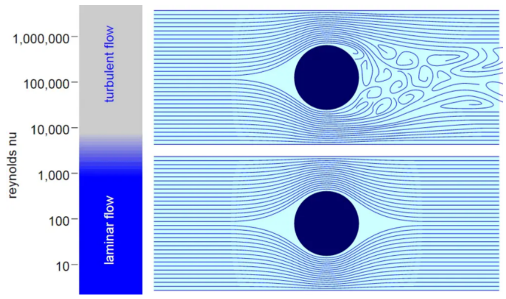


Figure 47: Laminar vs. turbulent flow

5.6.1 Subsonic (incompressible) turbulence, low Mach numbers | rotational modes

For subsonic turbulence, the information speed is far higher than the transport speed limiting the naturally occuring compressions (no shocks), so we can assume the fluid to be incompressible, $\underline{\nabla} \cdot \underline{v} = 0$.

In Fourier space the nabla operator becomes $\underline{\nabla} \rightarrow \underline{k}$, so $\underline{k} \cdot \underline{v} = 0$, so there are no longitudinal disturbances (aka soundwaves) but only solenoidal, i.e. source free motion, so shear flow and rotational turbulence (swirling eddies as for instance produces by the Kelvin-Helmholtz instability).

$\underline{v} \cdot \underline{v} = 0$ also implies subsonic flow as supersonic velocities would cause shocks coming with compression of the post-shock fluid.

Incompressible turbulence is described by Kolmogorov's theory of incompressible turbulence.

Incompressible flow is governed by the Navier-Stokes equation for an incompressible fluid, so

$$\partial_t \underline{v} + \overbrace{(\underline{v} \cdot \underline{\nabla})\underline{v}}^{\text{advective transport}} = \underline{g} \underbrace{-\frac{1}{\rho}\underline{\nabla} P}_{\text{pressure force}} + \overbrace{\nu \nabla^2 \underline{v}}^{\text{viscous dissipation}} \tag{218}$$

5.6.2 How to quantify turbulence? - Reynolds number

Let us come back to the Reynolds number characterizing the ratio of the advective to the friction term in the Navier-Stokes equation

$$\begin{gathered} Re = \frac{\text{advective term}}{\text{frictional term}} = \frac{|(\underline{v} \cdot \underline{\nabla})\underline{v}|}{\nu \underline{\nabla}^2 \underline{v}} = \frac{V_0 L_0}{\nu} \\ \text{with } V_0 = \text{characteristic velocity}, \quad L_0 = \text{characteristic length scale}, \\ \nu = \text{kinematic viscosity} = \frac{\eta}{\rho} \sim \lambda_{mfp} v_{th} \\ \text{with } \lambda_{mfp} = \text{mean free path}, \quad v_{th} = \text{thermal velocity} \end{gathered} \tag{219}$$

Although counterintuitive, the viscosity increases with larger mean free path, intuitively as shear stress information is transported over larger distances.

In this view a high Reynolds number means that turbulence is generated faster by the chaotic advective term than is descroyed via dissipation.

For approximately $Re > 3.5 \cdot 10^3$ turbulence is expected, the interstellar medium has $Re \sim 10^8$. Oceans (viscosity of water $\sim 10^{-6}\,\mathrm{m}^2\,\mathrm{s}^{-1}$) and atmosphere are always turbulent, except for boundary layers (with small characteristic length scale).

5.6.2.1 Reynolds number as the ratio between advection and dissipation timescale

Most simply by dimensional analysis, one can yield (ν in units of $\frac{m^2}{s}$, a diffusion coefficient)

$$t_{adv} = \frac{L_0}{V_0}, \quad t_{dis} = \frac{L_0^2}{\nu} \quad \rightarrow \quad Re = \frac{t_{dis}}{t_{adv}} = \frac{L_0 V_0}{\nu} \sim \frac{L_0 V_0}{\lambda_{mfp} v_{th}} \tag{220}$$

where a high Reynolds number means that advection is faster than dissipation, thus dissipation cannot stabilize the growth of turbulence sufficiently and we have turbulent flow. In the equation we can also see that the Reynolds number is the product of the macroscopic-to-microscopic length and velocity scales.

5.6.3 Supersonic turbulence, shocks $\mathcal{M} \gg 1$ | rotational and compressive modes

Depending on the dimensionality, we have

- 1D: 1 compressive mode
- 2D: 1 compressive mode, 1 solenoidal mode
- 3D: 1 compressive mode, 2 solenoidal modes

5.6.4 Schematic concept of turbulence

In 3D

- **injection range** energy is injected on macroscopic scales, typical scale L, velocity v
- **inertial range** large eddies break up into smaller eddies and energy is transferred to smaller scales, vorticity ($\underline{\zeta} = \underline{\nabla} \times \underline{v}$) is conserved and no energy is dissipated
- **dissipation range** at the microscopic viscous scale (λ_{visc}, roughly the mean free path λ_{mfp}) energy is dissipated into viscous heat

In 2D the energy flow is reverted from small to large (inverse cascade).

5.6.5 Kolmogorov scales of turbulence

In the following consider

$$\begin{gathered}\text{largest eddy scale: } L_S, \quad \text{dissipation scale: } L_k \\ \text{rate of energy dissipation on small scales, energy flow in the inertial range: } \epsilon \\ \text{some eddy size: } \lambda, \quad \text{velocity on that scale: } v_\lambda \\ \text{fluid viscosity: } \nu \text{ in } \frac{m^2}{s}\end{gathered} \tag{221}$$

5.6.5.1 Dissipation scale - smallest scale to be resolved in a simulation

Assume high Reynolds number and start at the small scale. At the small scale, turbulent motions are statistically isotropic (different from the large scales L_S) and Kolmogorov postulates the statistics to be universally determined by ν and ϵ (in $\mathrm{m}^2 / \mathrm{s}^3$). From dimensional analysis (physical units), one can then get

$$L_K \sim \left(\frac{\nu^3}{\epsilon}\right)^{\frac{1}{4}} \tag{222}$$

Note: This is the smallest scale to be resolved in a classical simulation, for an airplane with chord length $2\,\mathrm{m}$ this is $\mathcal{O}(10^{-6})\,\mathrm{m}$ - quite a problem for simulations.

5.6.6 Scaling of the eddy velocity and vorticity in the inertial range

Energy flow ϵ must be constant in the inertial range (otherwise accumulation). We approximate the energy flow by the kinetic energy of an eddy divided by its characeteristic time scale

$$\epsilon \approx \left(\frac{v_\lambda^2}{2}\right)\left(\frac{v_\lambda}{\lambda}\right) \approx \frac{v_\lambda^3}{\lambda} \underbrace{\approx}_{\epsilon=\text{const.}} \frac{v_{L_S}^3}{L_S} \tag{223}$$

therefore, we get

$$v_\lambda \approx v_{L_S}\left(\frac{\lambda}{L_S}\right)^{\frac{1}{3}} \tag{224}$$

Note: The largest eddies have highest velocities.

But as the size scales down quicker than the velocity, smallest eddies have the highest vorticity

$$|\underline{\zeta}_\lambda| \approx \frac{v_\lambda}{\lambda} \approx \frac{v_{L_S}}{(\lambda^2 L_S)^{\frac{1}{3}}} \tag{225}$$

Note: As the vorticity increases with decreasing scale but overall vorticity is approximately constant ($\sim$ conservation of angular momentum) on smaller scales less volume is filled with turbulent eddied.

5.6.7 Power spectrum of Kolmogorov turbulence

The dissipation is reflected by the energy spectrum of Kolmogorov turbulence, see figure 48.

The constant energy transfer through the cascade is described by

$$\begin{aligned} \text{incompressible fluid, subsonic:} \quad & E(k) \propto k^{-\frac{5}{3}} \\ \text{compressible, shock-dominated:} \quad & E(k) \propto k^{-2} \end{aligned} \tag{226}$$

Note: Including the the dissipation range, based on Kolmogorovs assumption that the statistics for small scale motion are universal, one might make the ansatz $E(k) = C\epsilon^{\frac{2}{3}}k^{-\frac{5}{3}}f_\eta(k\eta)$ with $f_\eta(x) = 1$ for $x \ll 1$ and $f_\eta(x) \to 0$ for $x \to \infty$.

5.6.7.1 Derivation of the energy spectrum of Kolmogorov turbulence

See Springel et al., 2023.

Maybe add derivation of Kolmogorov spectrum.

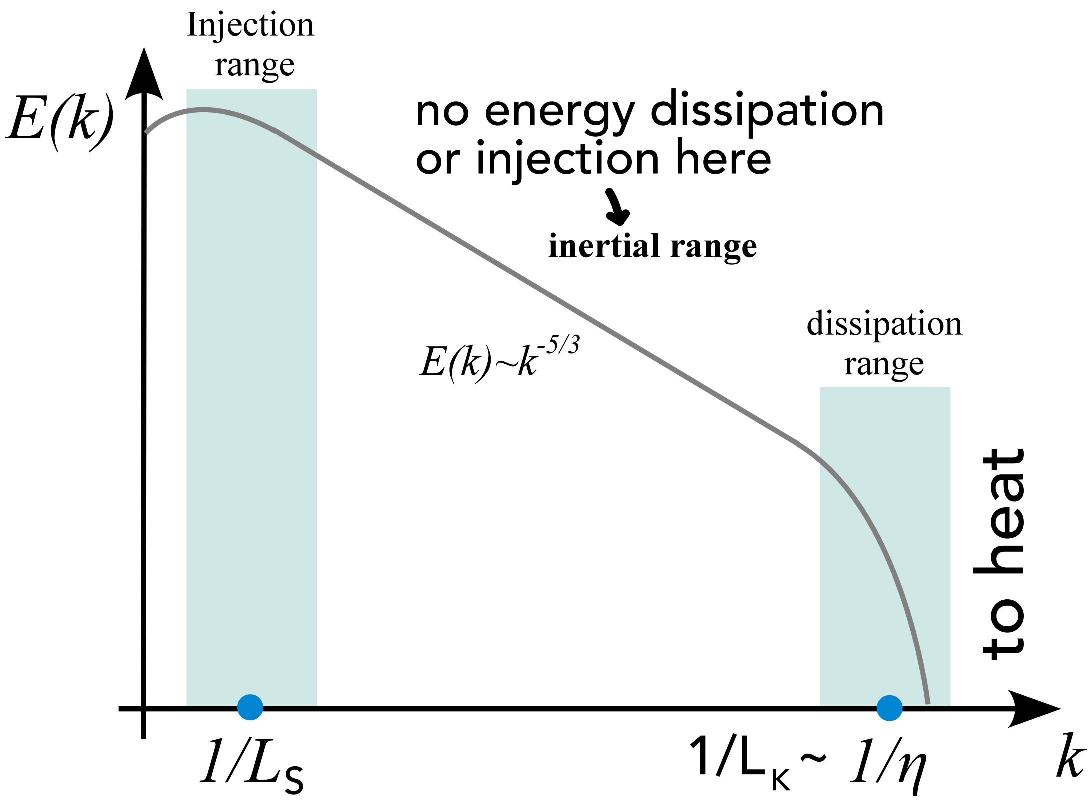


Figure 48: Kolmogorov energy spectrum

6 Eulerian Hydrodynamics | Solving PDEs

As our overarching aim is simulating physical systems (e.g. how do perturbations evolve in a fluid) and most physical systems are described by partial differential equations (PDEs) (like the Euler equations), we turn our heads towards numerically solving PDEs. We search for functions complying to $\mathcal{P}[u] = 0$ (and boundary conditions) with $\mathcal{P}$ being a differential operator.

Note: This section focuses on solvers following fluid variables on a fixed grid - eulerian hydrodynamics.

6.1 Introductory notes on PDEs

Partial differential equations (like Euler and Navier Stokes equation, Maxwell's equation, ...) describe relations between partial derivatives of a dependent variable (e.g. fluid density) with respect to several independent variables (e.g. time and position), like

$$\partial_t u = \partial_x^2 u, \quad \text{dependent variable } u(x,t), \quad \text{independent variables } x, t \tag{227}$$

(a kind of 1D diffusion equation).

Note: There is no general approach for solving PDEs.

6.2 Types of PDEs

PDEs are classified by

- Order of the PDE: Order of the highest occuring derivative
- Linearity: If the dependent variable and all its derivatives only occur linearly (no $\sqrt{\partial_x u}$), the PDE is linear and if u_1 and u_2 are solutions to the PDE, $c_1 u_1 + c_2 u_2$ are as well (superposition)
- Homogeneity: The PDE is homogeneous, if all terms contain the dependent variable or its derivatives, so if there is no source term

6.2.1 Classification of linear 2nd order PDEs in analogy with conic sections

PDEs can be distinguished into different types which give clues about appropriate solution strategies as well as appropriate initial and boundary conditions and the smoothness of the solution.

Consider a 2nd order linear PDE with two independent variables, so generally

$$a\frac{\partial^2 u}{\partial x^2} + b\frac{\partial^2 u}{\partial x \partial y} + c\frac{\partial^2 u}{\partial y^2} + d\frac{\partial u}{\partial x} + e\frac{\partial u}{\partial y} + fu = g, \quad a, b, c \text{ not all zero} \tag{228}$$

6.2.1.1 Derivation | homogeneous solutions are conic section in k-space

The unknown function u is expanded into plane waves (Fourier transformation)

$$u(\underline{x}) = \frac{1}{(2\pi)^2}\int \hat{u}(\underline{k})\exp(-i\underline{k}\cdot\underline{x})d^2k, \quad \underline{k} =: \begin{pmatrix} k \\ l \end{pmatrix} \tag{229}$$

Plugging this into the PDE yields

$$-\frac{1}{(2\pi)^2}\int \left(ak^2 + bkl + cl^2 + idk + idk - f\right)\hat{u}(\underline{k})\exp\left(-i\underline{k}\cdot\underline{x}\right)d^2k = g \tag{230}$$

For the homogeneous solution ($g = 0$) this must be zero, so

$$\underline{k}^T \underline{\underline{\Delta}}\,\underline{k} + i\begin{pmatrix} d \\ e \end{pmatrix}^T \underline{k} - f = 0, \quad \underline{\underline{\Delta}} = \begin{pmatrix} a & b/2 \\ b/2 & c \end{pmatrix} \tag{231}$$

which is the matrix representation of a conic section in $\underline{k}$-space (why?).

6.2.1.2 Classification into elliptic, parabolic, hyperbolic

$$\text{The PDE is} \begin{cases} \text{hyperbolic if } D > 0 \\ \text{parabolic if } D = 0, \\ \text{elliptic if } D < 0 \end{cases} \quad \Delta = \begin{vmatrix} a & b/2 \\ b/2 & c \end{vmatrix}, \quad D = -4\Delta = b^2 - 4ac \quad (232)$$

6.2.1.3 Qualitative differences on the types of PDEs

- Elliptic
  - often describe static problems (no time dependence) like the Poisson equation ($\underline{\nabla}^2\phi = -\frac{\varrho}{\epsilon}$)
  - as smooth as coefficients allow in the interior region where the equation and solutions are defined (independent of the smoothness of the boundary conditions)
- Parabolic
  - often of second order, describing slowly changing processes like diffusion, becoming smoother with time
  - the problem is described using an initial state $u(x, t_0)$ as well as boundary conditions
  - all parabolic PDEs can be transformed into a form analogous to the heat equation by change of independent variables
- Hyperbolic
  - typically describe dynamical processes in physics
  - initial conditions are specified by $u(x, t_0)$, $\frac{\partial u}{\partial x}(x, t_0)$ and higher derivatives as necessary as well as boundary conditions
  - disturbances have finite propagation speed
  - solutions can develop steep regions and real discontinuities

Boundary types: Fixed function values on the boundaries = Dirichlet boundary; fixed value of the derivative = van Neumann boundary.

Equation	Classification
Laplace equation $$\underline{\nabla}^2 u = \partial_x^2 u + \partial_y^2 u = 0 \quad (a=1, b=0, c=1) \qquad (233)$$	Elliptic
1D Heat conduction equation (diffusion equation) $$\partial_t u - \lambda^2 \partial_x^2 u = 0 \quad (a=0, b=0, d=1, c=-\lambda^2) \qquad (234)$$	Parabolic
1D Wave equation $$\partial_t^2 u - c_s^2 \partial_x^2 u = 0 \quad (a=1, b=0, c=-c_s^2) \qquad (235)$$	Hyperbolic

Table 10: Typical examples and classification of homogeneous 2nd order PDEs

6.2.2 Typical examples and classification of homogeneous 2nd order PDEs

Note: The classification scheme introduced is for 2nd order PDEs with two variables, it is not made e.g. for the advection equation $\partial_t u + v\partial_x u = 0$ which is first order and hyperbolic.

6.2.3 Classification of linear 2nd order PDEs with more unknowns

For the general from

$$\sum_{i,j=1}^{n} A_{ij} \frac{\partial^2 u}{\partial x_i \partial x_j} + \sum_{i=1}^{n} b_i \frac{\partial u}{\partial x_i} + cu + d = 0, \quad \underline{\underline{A}} \text{ with entries } A_{ij} \tag{236}$$

regarding the eigenvalues of $\underline{\underline{A}}$ (following from $\underline{\underline{A}}\underline{v} = \lambda \underline{v} \rightarrow \det(\underline{\underline{A}} - \lambda \underline{\underline{I}}) = 0$) it holds that

$$\text{The PDE is} \begin{cases} \text{hyperbolic if one negative, rest postitive or one positive, rest negative} \\ \text{parabolic if one zero, others all positive or all negative} \\ \text{elliptic if all positive or all negative} \end{cases} \tag{237}$$

Note that this classification is not exhaustive, the occurrence of multiple different signs is sometimes called ultra-hyperbolic.

Task to the reader: Check that for $n = 2$ this classification scheme is equivalent to the previous one.

6.2.4 Linear systems of 1st order homogeneous PDEs

Question: When is a 1st order homogeneous PDE hyperbolic?

A first order homogeneous PDE is of the form

$$\partial_t u_i + \sum_{j=1}^{n} A_{ij} \partial_{x_j} u_i = 0, \quad i = 1, \dots, n \tag{238}$$

with short-hand notation

$$\partial_t \underline{u} + \left(\underline{\underline{A}} \underline{\underline{\nabla}}_{\underline{x}} \right) \underline{u} = 0, \quad \underline{\underline{A}} \text{ with entries } A_{ij}, \quad \underline{\underline{\nabla}}_{\underline{x}} = \mathrm{diag}(\partial_{x_1}, \dots, \partial_{x_n}) \tag{239}$$

where **the system is hyperbolic if $\underline{\underline{A}}$ has real eigenvalues and is diagonizable**.

6.2.4.1 Extension to conservation laws

We extend this to conservation laws

$$\partial_t \underline{u} + \underline{\nabla}_{\underline{x}} \cdot \underline{\underline{F}}(\underline{u}) = \partial_t \underline{u} + (\underline{\underline{\nabla}}_{\underline{u}} \underline{\underline{F}}(\underline{u})) \underline{\underline{\nabla}}_{\underline{x}} \underline{u} = 0 \tag{240}$$

with conserved variable $\underline{u}$ and flux matrix $\underline{\underline{F}}(\underline{u})$. IF $\underline{\underline{\nabla}}_{\underline{u}} \underline{\underline{F}}(\underline{u})$ is diagonizable with real eigenvalues, the system is hyperbolic.

Consider e.g. the Navier-Stokes equation

$$\partial_t (\rho \underline{v}) + \underline{\nabla} \cdot (\rho \underline{v} \underline{v}^T + P \underline{\underline{I}} - \underline{\underline{\Pi}}) = \rho \underline{g} \tag{241}$$

where $\underline{\nabla} \cdot$ here is a matrix divergence (a vector) with entries as expected for such a matrix vector multiplication.

Note: The classification introduced has its limits: What type the Navier-Stokes equation is seems different to tell, often *hyperbolic* is used as *advection-dominated* (the advection equation is hyperbolic) and *parabolic* as *diffusion-dominated* (the diffusion equation is parabolic) and then the Navier stokes equation can be either depending on the Reynolds number.

6.3 Solution schemes for PDEs

There is no general approach but mutltiple general methods for different types or even only a certain PDE. Common methods are

- Finite difference methods: Differential operators are approximated by finite difference operators, usually on a regular (Cartesian) mesh
- Finite volume methods: Useful for hyperbolic conservation laws. We consider quantities averaged over finite volumes around mesh cells where divergence terms in a PDE turn into fluxes through the cells surface ($\rightarrow$ conservative method, fluxes are not lost). Exact expressions for the average value of the solution over some volumes are calculated and from these averages solutions within the cells can be reconstructed. This **contrasts with finite difference methods where derivatives are approximated based on nodal values and finite element methods where local approximations are made using local data** and stitched together to a global solution.
- Spectral methods: The PDE is converted into an algebraic form (e.g. by Fourier transform), the solution is represented by a linear combination of functions (e.g. the plane waves from the inverse Fourier transform).
- Method of lines: All derivatives but one are approximated by finite differences, leading to an ODE system, where we can use ODE solvers. **Time-dependent problems:** Consider a time-dependent problem on a 1D grid, $x_i, i = 1, ..., N$. Each point yields and ODE for the time evolution of the solution at that point, dependent on e.g. the solution on neighboring points $\rightarrow N$ coupled ODEs.
- Finite element methods: The simulation domain is divided into cells (elements, arbitrary shape, unstructured mesh) and the solution on each cell approximated in form of a simple (polynomial) function. The solutions from the cells are linearly combined and we solve for the coefficients.

Example for the method of lines: Consider the 1D heat equation

$$\partial_t u - \lambda^2 \partial_x^2 u = 0 \tag{242}$$

How to discretize the 2nd order derivative $\partial_x^2 u$?: Second order Taylor expansion yields

$$\begin{aligned} u(x+\Delta x) &= u(x) + \Delta x \partial_x u(x) + \frac{1}{2}\Delta x^2 \partial_x^2 u(x) + \mathcal{O}(\Delta x^3) \\ u(x-\Delta x) &= u(x) - \Delta x \partial_x u(x) + \frac{1}{2}\Delta x^2 \partial_x^2 u(x) + \mathcal{O}(\Delta x^3) \end{aligned} \tag{243}$$

where adding both yields

$$\partial_x^2 u(x) \approx \frac{u(x+\Delta x) - 2u(x) + u(x-\Delta x)}{\Delta x^2} \tag{244}$$

We therefore get

$$\partial_t u_i - \lambda^2 \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} = 0, \quad i = 1, \ldots, N \tag{245}$$

which we could for instance approach using the Euler method. The grid is illustrated in fig. 49.

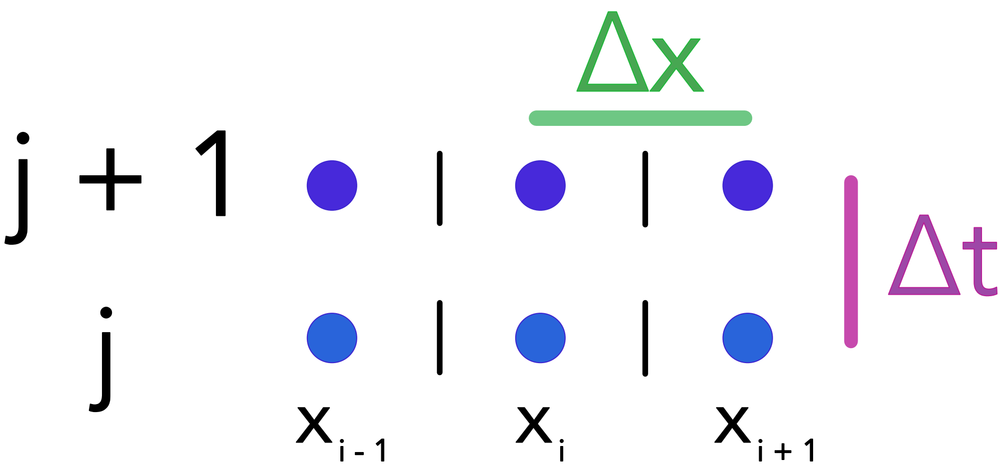


Figure 49: 1D grid

Problem: This scheme might not be stable.

6.4 Advection - Keep information flow in the physical system in mind

Consider the first-order hyperbolic advection equation

$$\partial_t u + v\partial_x u = 0, \text{we seek } u(x,t), \quad v \text{ constant parameter} \tag{246}$$

which is hyperbolic as of the real and *diagonizable* coefficient matrix (a scalar).

6.4.0.1 Analytic solution to the advection equation

For any function $q(x)$, the function $u(x,t) = q(x - vt)$ is a solution to the advection equation ($\partial_t u = -v\partial_x q$) (see fig. 50).

Interpreting $u(x, t = 0) = q(x)$ as an initial condition, the solution at a later time is just a shifted (by vt along x) copy of $q(x)$.

While for the advection problem we know the analytic solution, it helps us uncover the basic cavets of numerically solving PDEs.

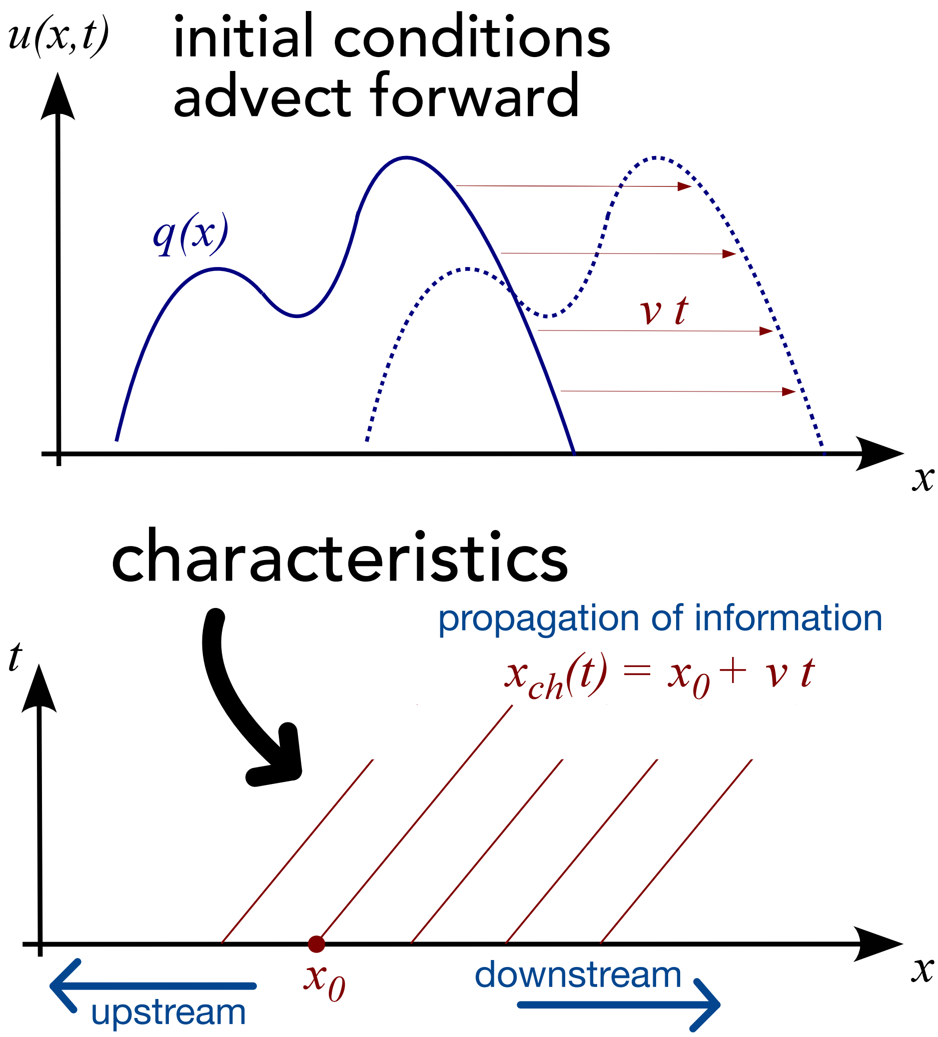


Figure 50: Advection

6.4.0.2 Simple but wrong approach | we need to consider the flow of information

Let us replace the derivative in space by a central difference with respect to the neighboring grid points

$$\frac{\partial u_i}{\partial t} + v\frac{u_{i+1} - u_{i-1}}{2h} = 0 \tag{247}$$

and step in time using explicit Euler

$$u_i^{(n+1)} = u_i^{(n)} - v\frac{u_{i+1}^{(n)} - u_{i-1}^{(n)}}{2h}\Delta t \tag{248}$$

Problem: This is violently unstable (illustrated in 51), as (based on the characteristics) information should only travel downstream but in the central differencing, we use upstream information $u_{i+1}^{(n)}$

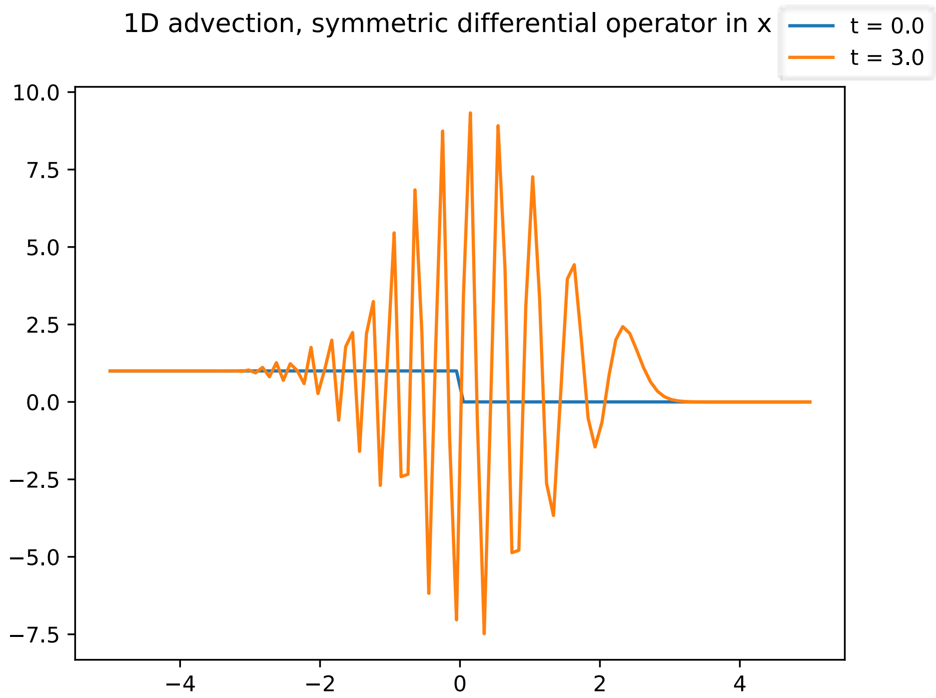


Figure 51: Advection with central differencing, violently unstable

6.4.0.3 Directional splitting / upwind scheme to the rescue

Let us only use downstream information, so (mind the sign of v)

$$\begin{aligned} v > 0 : u_i^{(n+1)} &= u_i^{(n)} - v\frac{u_i^{(n)} - u_{i-1}^{(n)}}{h}\Delta t \\ v < 0 : u_i^{(n+1)} &= u_i^{(n)} - v\frac{u_{i+1}^{(n)} - u_i^{(n)}}{h}\Delta t \end{aligned} \tag{249}$$

An example application is given in fig. 52.

Problem: The solution is smeared out (smoothed).

6.4.0.4 Where does the smoothing in the upwind scheme come from?

Our numerical algorithm (that lead to stability) introduced numerical diffusion as a byproduct.

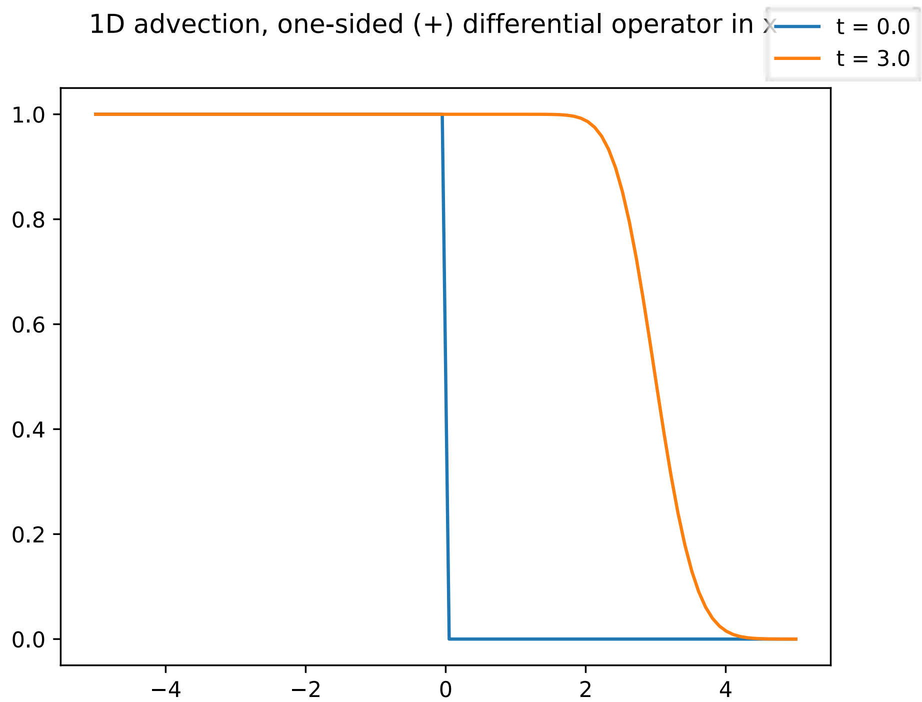


Figure 52: Advection with upwind scheme

Using

$$\frac{u_i - u_{i-1}}{h} = \frac{u_{i+1} - u_{i-1}}{2h} - \frac{u_{i+1} - 2u_{i+1} + u_{i-1}}{2h} \tag{250}$$

we can rewrite the upwind scheme for $v > 0$ as

$$\begin{aligned} 0 = \partial_t u_i + v\frac{u_i - u_{i-1}}{h} &= \partial_t u_i + v\frac{u_{i+1} - u_{i-1}}{2h} - \underbrace{\frac{vh}{2}}_{D} \underbrace{\frac{u_{i+1} - 2u_{i+1} + u_{i-1}}{h^2}}_{\text{discretization of } \partial_x^2 u} \\ \text{so } \partial_t u_i + v\frac{u_i - u_{i-1}}{h} &= D\frac{u_{i+1} - 2u_{i+1} + u_{i-1}}{h^2} \end{aligned} \tag{251}$$

which is the the central difference version of a advection-diffusion equation

$$\partial_t u + v\partial_x u = D\partial_x^2 u \tag{252}$$

where the diffusion term smears out the solution.

The numerical diffusion term $D = \frac{vh}{2}$ is small for

- fine grids ($h \to 0$)
- small velocities ($v \to 0$), stronger advection $\to$ stronger diffusion

The diffusion term dampens all post-shock oscillations / oscillations connected to steep gradient and can thus also be useful for stabilization.

6.4.0.5 What is the maximum timestep we can take? | Courant-Friedrichs-Lewy (CFL) criterion

Consider the advection problem. Information travels with velocity v. Consider you would take a timestep $\Delta t > \frac{h}{v}$. Then in the upwind scheme, we would not only need to consider u_{i-1} (assuming $v > 0$) but also u_{i-2}, which we do not do - leading to catastrophic instability, as illustrated in fig. 53.

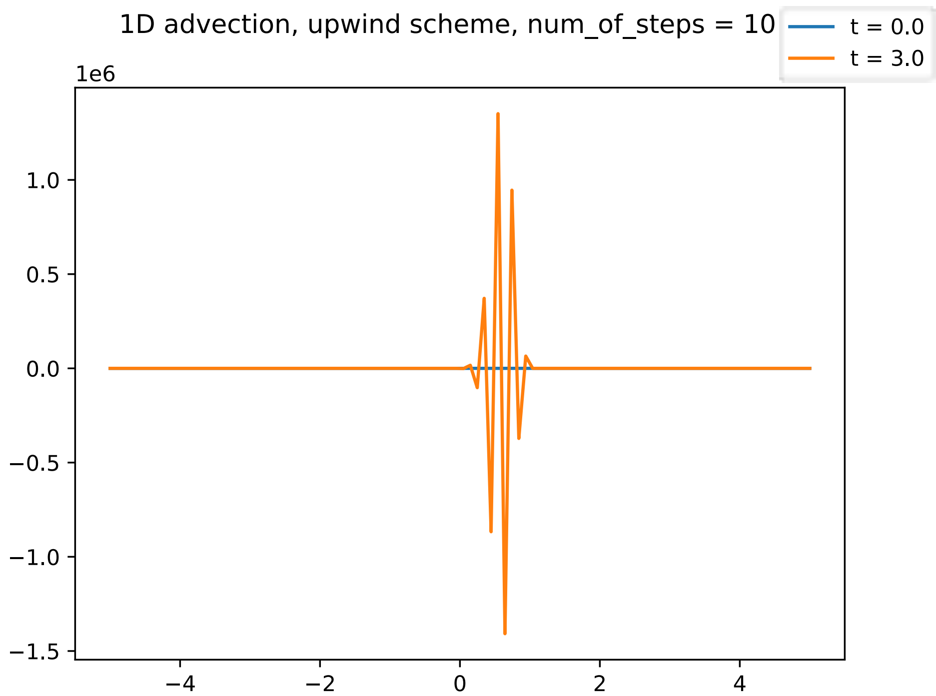


Figure 53: Advection with upwind scheme, violating the CFL criterion

The CFL criterion therefore reads

$$\Delta t \leq \frac{h}{v} \tag{253}$$

a necessary but not suffiecient condition for the stability of explicit methods regarding hyperbolic conservation laws (here for the advection case, might generally take different forms)

Note: Integrating hyperbolic conservation laws in time needs sufficiently small integration steps, as there is a finite speed of information travel in such hyperbolic problems.

6.4.0.6 Hyperbolic conservation laws | changing upwind direction

Consider the continuity equation

$$\partial_t \rho + \underline{\nabla} \cdot \underline{F} = 0, \quad \text{mass flux } \underline{F} = \rho \underline{v} \tag{254}$$

While this is essentially an advection problem, $\underline{v}$ can vary over space, $\underline{v} = \underline{v}(\underline{x})$.

In a naive discretization of space and time

$$\frac{\rho_i^{(n+1)} - \rho_i^{(n)}}{\Delta t} + \frac{F_{i+1}^{(n)} - F_{i-1}^{(n)}}{2\Delta x} = 0 \to \rho_i^{(n+1)} = \rho_i^{(n)} + \frac{\Delta t}{2\Delta x}\left(F_{i+1}^{(n)} - F_{i-1}^{(n)}\right) \tag{255}$$

the solution is highly unstable as of not accounting for the direction of flow of information (here flow of mass). We need to choose the correct discretization depending on the direction of the characteristic / sign of mass flux.

6.4.0.7 What if identifying the local characteristics is very difficult?

In general (non-linear PDE) situations, information about the local solution and local characteristics is obtained using Riemann solvers.

6.5 Intermezzo: CFL like criterion and connection to stiffness in a reaction diffusion system

Introduction to the example problem - the Brusselator

As our example, we use a simplified *Brusselator* as introduced in Hairer, Wanner, and Nørsett, 1993, chapter I.16 and further discussed in Hairer and Wanner, 1996, chapter IV.1 (originally introduced in Lefever and Nicolis, 1971).

Some details on the Brusselator and all of our implementations regarding numerical methods for solving it can be found in the accompanying Julia notebook.

Here, it is sufficient to know that we consider a non-linear chemical-reaction-diffusion partial differential equation in one dimension of the form

$$\begin{aligned}
\frac{\partial u}{\partial t} &= A + u^2 v - (B+1)u + \alpha \frac{\partial^2 u}{\partial x^2} \\
\frac{\partial v}{\partial t} &= Bu - u^2 v + \alpha \frac{\partial^2 v}{\partial x^2}
\end{aligned}$$

where $u(x,t)$ and $v(x,t)$ are the concentrations of chemical substances, α is a diffusion constant and A and B are fixed concentrations of other substances.

From discretizing the differentiation in space (i.e. using the method of lines for approaching this partial differential equations) we follow (with $x_i = \frac{i}{N+1}(1 \leq i \leq N), \Delta x = \frac{1}{N+1}$, $A = 1$, $B = 3$, $\alpha = \frac{1}{50}$)

$$\begin{aligned} u_i' &= 1 + u_i^2 v_i - 4u_i + \frac{\alpha}{(\Delta x)^2} \left(u_{i-1} - 2u_i + u_{i+1}\right), \\ v_i' &= 3u_i - u_i^2 v_i + \frac{\alpha}{(\Delta x)^2} \left(v_{i-1} - 2v_i + v_{i+1}\right) \\ u_0(t) &= u_{N+1}(t) = 1, \quad v_0(t) = v_{N+1}(t) = 3 \\ u_i(0) &= 1 + \sin\left(2\pi x_i\right), \quad v_i(0) = 3, \quad i = 1, \ldots, N. \end{aligned} \tag{256}$$

where some boundary conditions and initial conditions have been chosen. The constant boundary values are enforced using so-called ghost-cells in the implementation.

We compactly write the differential equation system as

$$\underline{y} = \underline{f}\left(\underline{y}\right), \quad \underline{y} = \begin{pmatrix} u_0 \\ \vdots \\ u_{N+1} \\ v_0 \\ \vdots \\ v_{N+1} \end{pmatrix}$$

where $\underline{f} : \mathbb{R}^{2N} \to \mathbb{R}^{2N}$ follows from equation 256.

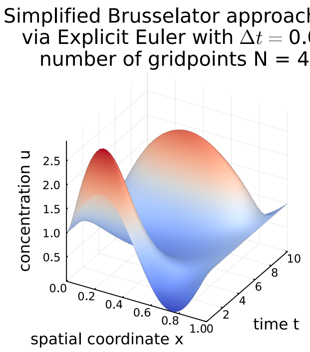
The occurrence of stiffness

Let us start by applying the Explicit Euler method to the simplified Brusselator with $N = 40$ grid points and a step size $dt = 0.01$. The result is shown in figure 54a and is in agreement with the literature results (Hairer and Wanner, 1996, chapter IV.1).

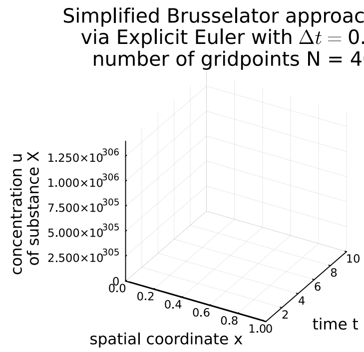
But if we increase N to $N = 400$, i.e. we decrease the spacing between the grid points $\Delta x = \frac{1}{N+1}$, the Explicit Euler scheme yields a diverging result (see figure 54b).

We can get back to a stable solution by decreasing the step size but notice that we have to use a much smaller step size, e.g. $dt = 0.0001$ (see figure 54c), in spite of the solution still being very smooth.

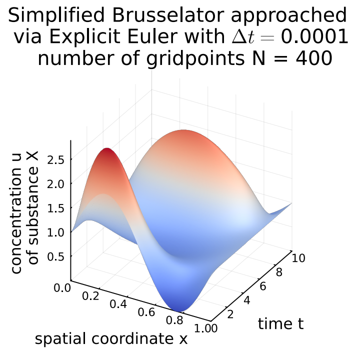
In fact even a more sophisticated explicit method like the Tsitouras 5/4 Runge-Kutta method from the DifferentialEquations.jl package (Rackauckas and Nie, 2017) will use excessively many steps (e.g. to cover a time interval of length 10 (dimensionless as of our problem



(a)



(b)



(c)

Figure 54: Numerical solutions to a simplified Brusselator using the Explicit Euler method with different numbers N of grid points and time-steps Δt.

formulation) using the default settings 220047 evaluations of $\underline{f}$ are used). The problem of stiffness as described in section 3.4.2 has occurred.

Understanding stiffness in a diffusive context

The transport process at play is diffusion. For a diffusive process we know the spreading of some concentration to follow $\sigma = \sqrt{2\alpha t}$. Now in each Euler step we do, only neighboring cells have an effect on each other (compare the discretized ODE we introduced in the beginning). Therefore - in the style of a Courant-Friedrichs-Levy criterion (Courant et al., 1928) - we can propose the stability constraint

$$\Delta x > \sigma(\Delta t) = \sqrt{2\alpha t}$$

so $\Delta t < \frac{\Delta x^2}{2\alpha}$. If we want to double N (cut in half Δx) we need $\mathcal{O}(N^2)$ more time-steps with the complexity of a function evaluation scaling with $\mathcal{O}(N)$ resulting in a $\mathcal{O}(N^3)$ scaling - calculations quickly become unfeasible.

Let us note that in the simplified Brusselator at hand it is the diffusive term causing stiffness, but in a more complex model the chemical reactions could be an additional factor of stiffness (see e.g. Chou et al., 2007).

6.6 Riemann problem | Riemann solvers

Consider a hyperbolic system. At time $t = 0$ we start out with two piecewise constant states (in the fluid variables) meeting at a plane. The Riemann problem is to determine the subsequent evolution.

Note: One reason the Riemann problem is important is that when we discretize a fluid into cells with constant values, we effectively have Riemann problems in-between.

Consider the Riemann problem for the Euler equations (ideal gas dynamics). The two constant states can be uniquely described by

$$\underline{U}_L = \begin{pmatrix} \rho_L \\ P_L \\ \underline{v}_L \end{pmatrix}, \quad \underline{U}_R = \begin{pmatrix} \rho_R \\ P_R \\ \underline{v}_R \end{pmatrix}, \quad \text{mass density } \rho, \quad \text{pressure } P, \quad \text{velocity } \underline{v} \tag{257}$$
$$\text{alternative primitive variables: density, momentum density, energy density}$$

with a hydrodynamic discontinuity in between (illustrated in 55). This can be solved analytically (but not be written down explicitly, requires numerical root-finding for an implicit equation).

The shock tube (for $\underline{v}_L = \underline{v}_R = 0$) is a common test for Riemann solvers. There is no smooth information transport across the shock (but many collisions) and hydrodynamics breaks down.

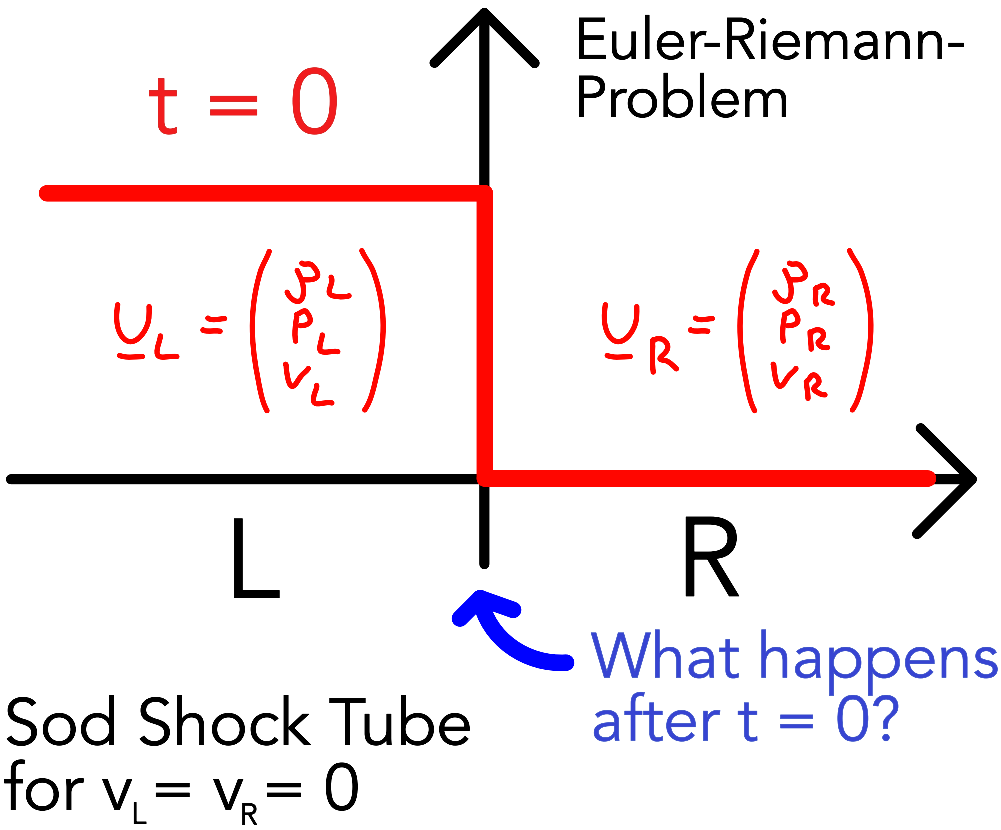


Figure 55: Riemann problem

6.6.1 Structure of the solution of the Euler-Riemann-Problem

In general, the solution to an Euler-Riemann problem always contains three waves

- **contact wave / discontinuity**: a middle wave marking the boundary between the original fluid phases
- on either side of the contact wave there can either be a **shock** or **rarefaction wave** (rather rarefaction fan with continuously changing variables). Shock or rarefaction on both sides is also possible.

where all three waves propagate with constant speed. At $x = 0$ the fluid quantities (ρ, P, v) (in the region containing the interface) are constant in time for $t > 0$.

6.6.1.1 Characteristics of the three waves

The characteristics are shown in fig. 56.

6.6.1.2 Example Riemann-Problem situation

An example with marked rarefaction fan, contact discontinuity and shock is shown in fig. 57.

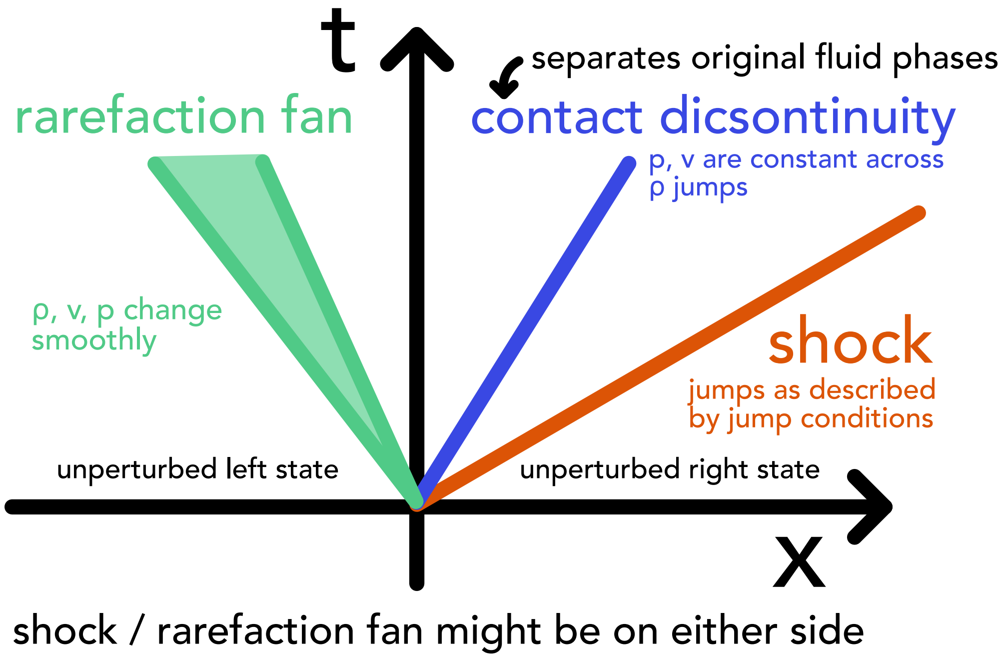


Figure 56: Characteristics of the three waves

6.6.1.3 Properties of shock, contact discontinuity and rarefaction wave

- Shock: Normal velocity, pressure, denseity, entropy change discontinuously. In the rest frame of the shock fast upstream fluid is converted to slow downstream one. The fluid is compressed and kinetic energy tuned into heat (addition of entropy).
- Contact discontinuity: Traces the original seperating plane between the two originally seperated fluid phases.
  - constant across the contact: pressure, normal velocity
  - can jump: density, entropy, temperature
- Rarefaction wave: smooth transition between two states, no discontinuities

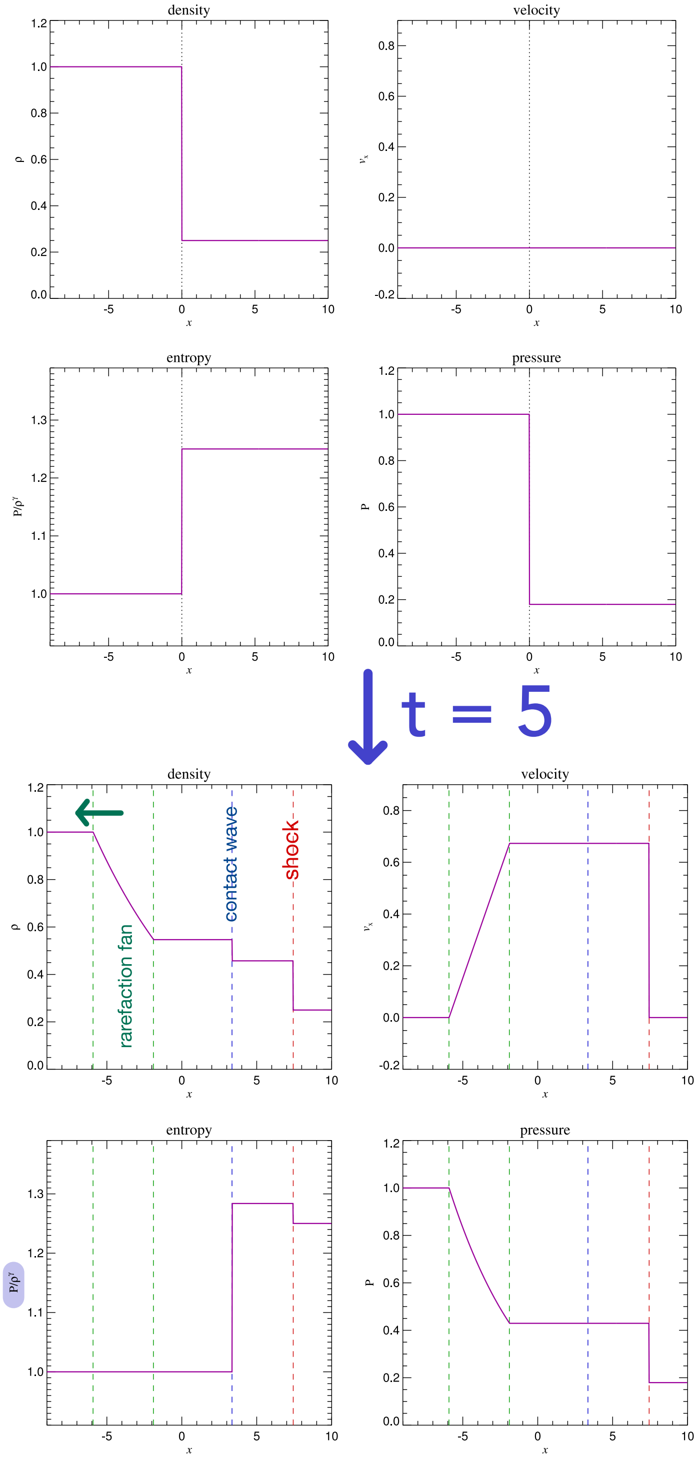


Figure 57: Example Riemann-Problem situation

6.7 Finite volume discretization | Reducing a hyperbolic conservation law to a Riemann problem | Godunov scheme

Idea: If we discretize the fluid into finite volumes and assume constant fluid variables on them, at each cell interface we have a Riemann problem. We are now more concerned with cell boundaries than centers. In the - conservative, finite volume - Godunov method, exact or approximate Riemann problems are solved at the boundaries and no flux is lost.

6.7.1 Problem | solve a hyperbolic conservation law PDE

We want to solve the conservation law

$$\partial_t \underline{U} + \partial_{\underline{x}} \cdot \underline{\underline{F}}(\underline{U}) = 0, \quad \text{state vector } \underline{U}, \quad \text{flux matrix } \underline{\underline{F}} \tag{258}$$

for instance for the Euler equations with

$$\underline{U} = \begin{pmatrix} \rho \\ \rho \underline{v} \\ \rho e \end{pmatrix}, \quad \underline{\underline{F}} = \begin{pmatrix} \rho \underline{v} \\ \rho \underline{v}\underline{v}^T + P\underline{\underline{1}} \\ (\rho e + P)\underline{v} \end{pmatrix}, \quad e = e_{th} + \frac{\underline{v}^2}{2}, \quad P = (\gamma - 1)\rho e_{th} \text{ closure} \tag{259}$$

6.7.2 Deriving a finite volume scheme where only Riemann problems are left to solve

In a finite volume method, the state of the cell is an average over the fluid quntities over the cell

$$\underline{U}_i = \frac{1}{V_i} \int_{\text{cell}i} \underline{U}(\underline{x}) dV \tag{260}$$

Aim: We want to derive an update scheme for the cell averages $\underline{U}_i$ (the vector of the fluid variables), where no intercell flux is lost.

We derive the update scheme in 1D, so $\underline{F}$ is a vector ($\underline{v}\underline{v}^T$ is a scalar) (see figure 58).

Step 1: Integrate the conservation law over a cell and timestep and recognize the average defined in 260.

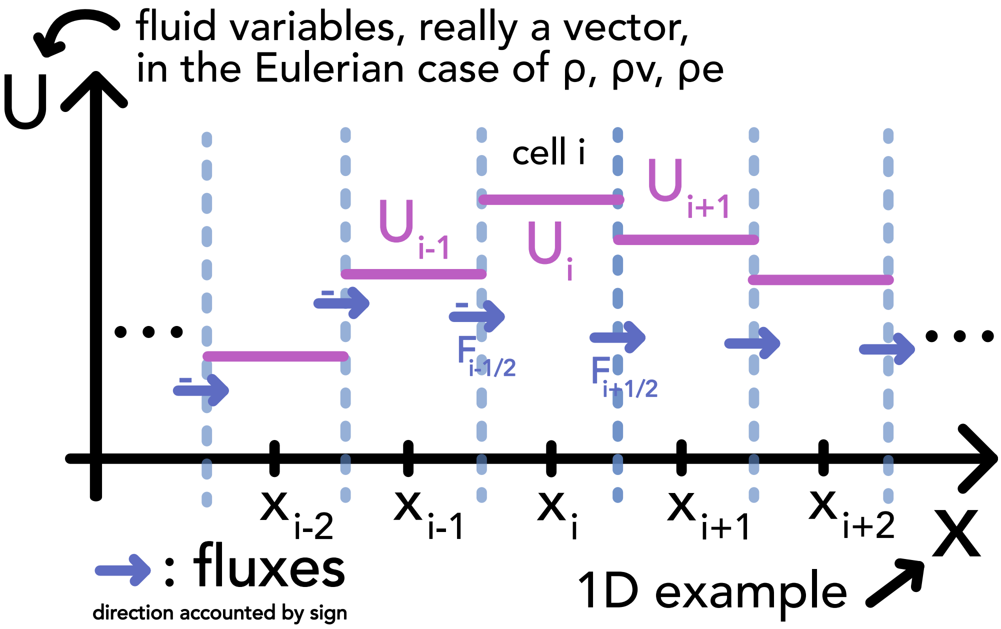


Figure 58: Finite volume scheme

$$
\begin{aligned}
&\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \int_{t_n}^{t_{n+1}} \left(\frac{\partial \underline{U}}{\partial t} + \frac{\partial \underline{F}}{\partial x} \right) dt dx = 0 \\
\underbrace{=}_{\text{carry out simple integrals}} &\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \left[\underline{U}\left(x, t_{n+1}\right) - \underline{U}\left(x, t_n\right) \right] dx + \int_{t_n}^{t_{n+1}} \left[\underline{F}\left(x_{i+\frac{1}{2}}, t\right) - \underline{F}\left(x_{i-\frac{1}{2}}, t\right) \right] dt \\
\underbrace{=}_{\text{recognize avg}} &\Delta x \left[\underline{U}_i^{(n+1)} - \underline{U}_i^{(n)} \right] + \int_{t_n}^{t_{n+1}} \left[\underline{F}\left(x_{i+\frac{1}{2}}, t\right) - \underline{F}\left(x_{i-\frac{1}{2}}, t\right) \right] dt = 0
\end{aligned}
\tag{261}
$$

Step 2: In the frame of discrete cell averages, between any two cells we essentially have a Riemann problem from which solution we can follow the flux between the cells.

$$
\underline{F}(x_{i+\frac{1}{2}}, t) \text{ for } t > t_n = \text{solution of Riemann problem with left state } \underline{U}_i^{(n)} \text{ and right state } \underline{U}_{i+1}^{(n)} \tag{262}
$$

Note: At the cell interface, the solution of the Riemann problem is constant in time, so

$$
\underline{F}(x_{i+\frac{1}{2}}, t) = \underline{F}^*_{i+\frac{1}{2}} = \underline{F}_{\text{Riemann}}(\underline{U}_i^{(n)}, \underline{U}_{i+1}^{(n)}) \text{ from Riemann solution at interface} \tag{263}
$$

Step 3: As the solution at the interface is constant, without approximation, we can write the Godunov scheme (based on eq. 261) as

$$\underline{U}_i^{(n+1)} = \underline{U}_i^{(n)} + \frac{\Delta t}{\Delta x}\left[\underbrace{\underline{F}^*_{i-\frac{1}{2}}}_{\text{flux from left into cell}} - \underbrace{\underline{F}^*_{i+\frac{1}{2}}}_{\text{out on the right}}\right] \tag{264}$$

This defined an update scheme for the fluid variables on the cells (with some appropriate initial and boundary conditions). $\underline{F}^*_{i-\frac{1}{2}}$ and $\underline{F}^*_{i+\frac{1}{2}}$ are black-boxes for now - to find the fluxes we need to solve the Riemann problem, later e.g. done with the approximate HLL solver.

6.7.2.1 Caveats of the Godunov scheme

CFL needs to be obeyed: We can only assume the Riemann problems at the interfaces to be independent, if the timestep is short enough, so that no information has travelled from one interface to the other → CFL: $\frac{\Delta x}{\Delta t} \leq c_{max}$.

$\underline{U}$ is not piecewise constant in reality: Even if we start out with piecewise constant $\underline{U}$ in reality, this will change. Therefore the flux we calculate between the cells is also only an approximation.

6.7.3 Godunov's method and Riemann solver | reconstruct - evolve - average (REA)

Godunov's method can be seen as a REA scheme of a hydrodynamical system discretized on a mesh

1. Reconstruct: A global solution is constructed from the cell averaged quantities, simples approach: piecewise-constant
2. Evolve: The reconstructed state is evolved by Δt (mind the CFL criterion), in the Gudonov scheme based on the intercell Riemann problems
3. Average: $\underline{U}_i^{(n+1)}$ is calculatted from the evolved state, in the Godunov scheme, evolving and averaging are combined as we directly calculate the new average based on accounting the fluxes entering and leaving the cell (an implicit average over the new state)

But how can the Riemann problem giving us the fluxes be solved?

6.8 Approximate Riemann solvers | HLL solver

6.8.1 1D Riemann problem to solve

Let us again formulate the Riemann problem in 1D. Given the conservation law

$$\partial_t u(x) + \partial_x f(u) = 0 \tag{265}$$

with piecewise initial values

$$u(x, t=0) = \begin{cases} u_L \text{ for } x < 0 \\ u_R \text{ for } x \geq 0 \end{cases}, \quad \text{cell interface at } x = 0 \tag{266}$$

we want to solve for our fluid quantity u. The characteristics, i.e. where information from point $x = 0$ (discontinuity) can travel in some time, are shown in figure 59.

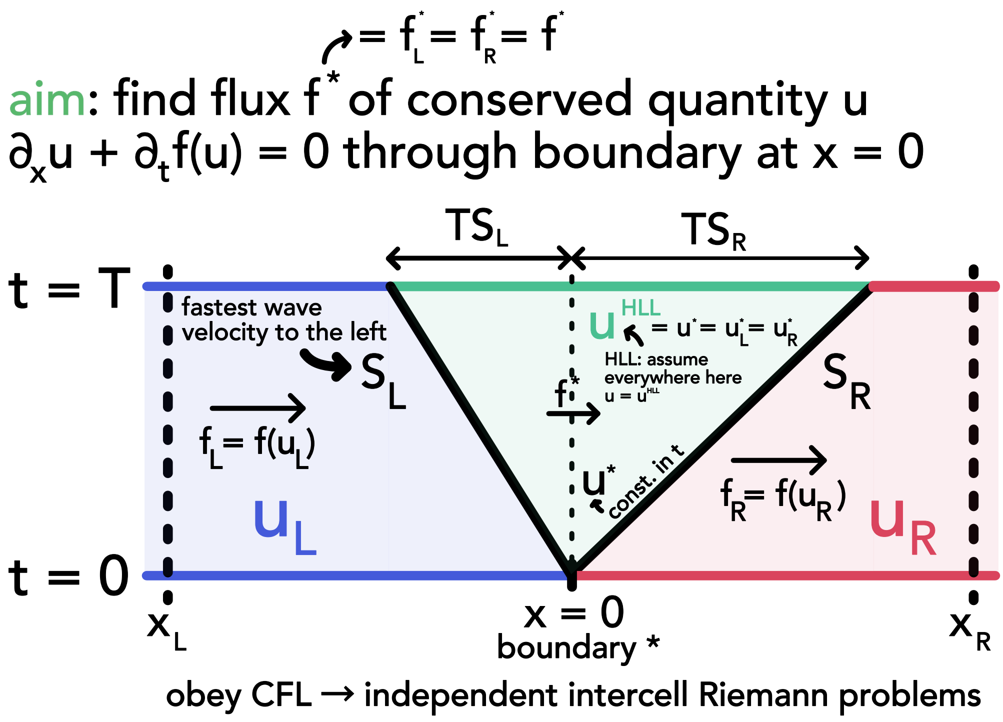


Figure 59: Characteristics of the Riemann problem and HLL approach.

6.8.2 Basic HLL assumptions and problem statement

In the HLL scheme we assume to know

- the fastest moving wave velocity to the left S_L and right S_R
- the left and right state u_L and u_R and therefore the fluxes f_L and f_R

Based on the CFL criterion (no interaction with other cells), we can assume the following quantities to be constant in time $t \in [0, T]$:

$$u_L = u(x_L, t), \quad u_R = u(x_R, t), \quad f_L = f(u_L), \quad f_R = f(u_R) \tag{267}$$

and we make the simplifying assumption that

$$\forall x \in [S_R t, S_L t] : u(x,t) = u^{HLL} = u_L^* = u_R^* = u^* \underbrace{=}_{\text{at interface}} \text{const.}, \quad f^{HLL} = f^* = f_L^* = f_R^* \tag{268}$$

Aim: Find expressions for u^* and f^*.

6.8.3 Deriving the solution of the Riemann problem in the HLL scheme

Idea: u is conserved, so the spatial integral over u at some point in time is the same as at another point in time plus / minus the in- and outcoming fluxes during that time-interval. Based and the flux balance in our whole region $[x_L, x_R]$ we can first find an expression for u^{HLL} and then based on the same balance in the left and right region $[S_R T, 0]$ and $[0, S_L T]$ we can find an expression for f^{HLL}.

6.8.3.1 Derivation of the middle state u^{HLL} at $t = T$

Remember, we assume the middle state u^{HLL} hold for the whole interval $[S_R T, S_L T]$, where information could have spread.

We can therefore write

$$u^{HLL} = \frac{1}{T \cdot (S_R - S_L)} \int_{TS_R}^{TS_L} u(x,T)dx \tag{269}$$

we can find this integral from considering the spatial integral over the whole domain at time T

$$\begin{aligned}
\int_{x_L}^{x_R} u(x,T)dx &= (TS_L - x_L)\, u_L + \int_{TS_R}^{TS_L} u(x,T)dx + (x_R - TS_R)\, u_R \\
&\underbrace{=}_{u \text{ conserved}} \int_{x_L}^{x_R} u(x,0)dx + \underbrace{\int_0^T f\left(u_L\right) dt}_{\text{flux in from left}} - \underbrace{\int_0^T f\left(u_R\right) dt}_{\text{flux out to right}} \\
&= u_R x_R - u_L x_L + f_L T - f_R T
\end{aligned} \tag{270}$$

We therefore find an expression for our integral and thus for u^{HLL}:

$$u^{HLL} = \frac{S_R u_R - S_L u_L + f_L - f_R}{S_R - S_L} \tag{271}$$

Which is a pretty simple balance we could have seen directly.

6.8.3.2 Deriving the intercell flux $f^{HLL} = f^*$

We consider the integral over u in the left region where the information can have propagated to, so $x \in [S_R T, 0]$ and in the right region $x \in [0, S_L T]$.

Idea: As before the integral of u over space at some time t must be the same as at an earlier time plus / minus the fluxes in / out since then.

For the left ($x \in [S_R T, 0]$) we have

$$\begin{aligned} -TS_L u^{HLL} &= \int_{TS_L}^{0} u(x,T)dx \\ &= \underbrace{-TS_L u_L}_{\text{at t=0}} + \underbrace{T \cdot (f_L - f_L^*)}_{\text{fluxes since then}} \end{aligned} \tag{272}$$

For the right ($x \in [0, S_L T]$) we have

$$\begin{aligned} TS_R u^{HLL} &= \int_{0}^{TS_R} u(x,T)dx \\ &= \underbrace{TS_R u_R}_{\text{at t=0}} + \underbrace{T \cdot (f_R^* - f_R)}_{\text{fluxes since then}} \end{aligned} \tag{273}$$

The fluxes at the interface must be equal, $f_L^* = f_R^* = f^{HLL}$, so

$$f^{HLL} = f_L + S_L \left(u^{HLL} - u_L\right) = f_R + S_R \left(u^{HLL} - u_R\right) \tag{274}$$

6.8.4 Final HLL solution

There are three states after the step T in the HLL scheme

$$u(x,T) = \begin{cases} u_L \text{ for } x < S_L t \\ u^{HLL} \text{ for } S_L t \leq x \leq S_R t \\ u_R \text{ for } x > S_R t \end{cases} \tag{275}$$

and the interface flux is

$$
\begin{gathered}
f^{HLL} = \frac{S_R f_R - S_L f_L + S_L S_R \left(u_R - u_L\right)}{S_R - S_L} \\
\text{maximum velocity to the left } S_L, \quad \text{maximum velocity to the right } S_R \\
\text{initial state on the left } u_L, \quad \text{initial state on the right } u_R \\
f_L = f(u_L), \quad f_R = f(u_R)
\end{gathered} \tag{276}
$$

6.8.5 Mind that the extreme velocities can point into the same direction

Both extreme velocities can be to the left, both to the right, or on to the left and one to the right. For instance in figure 60 characteristics for advection are drawn.

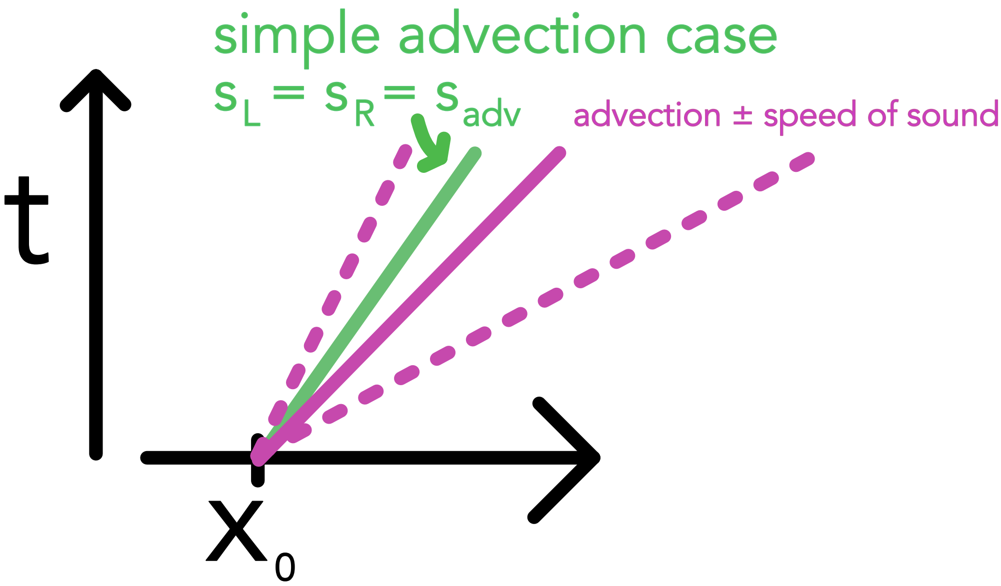


Figure 60: Advection characteristics, once the simple advection case with only the advection speed and once with information travelling from the advected state to the left and right with the speed of sound.

6.8.6 Godunov scheme with HLL solver

For the Godunov scheme

$$
U_i^{(n+1)} = U_i^{(n)} + \frac{\Delta t}{\Delta x}\left[F^*_{i-\frac{1}{2}} - F^*_{i+\frac{1}{2}}\right] \tag{277}
$$

the flux we choose depends on the orientation of S_L and S_R (eveloping the *Lichtkegel*). This is illustrated for unidirectional information flow to the right in figure 61.

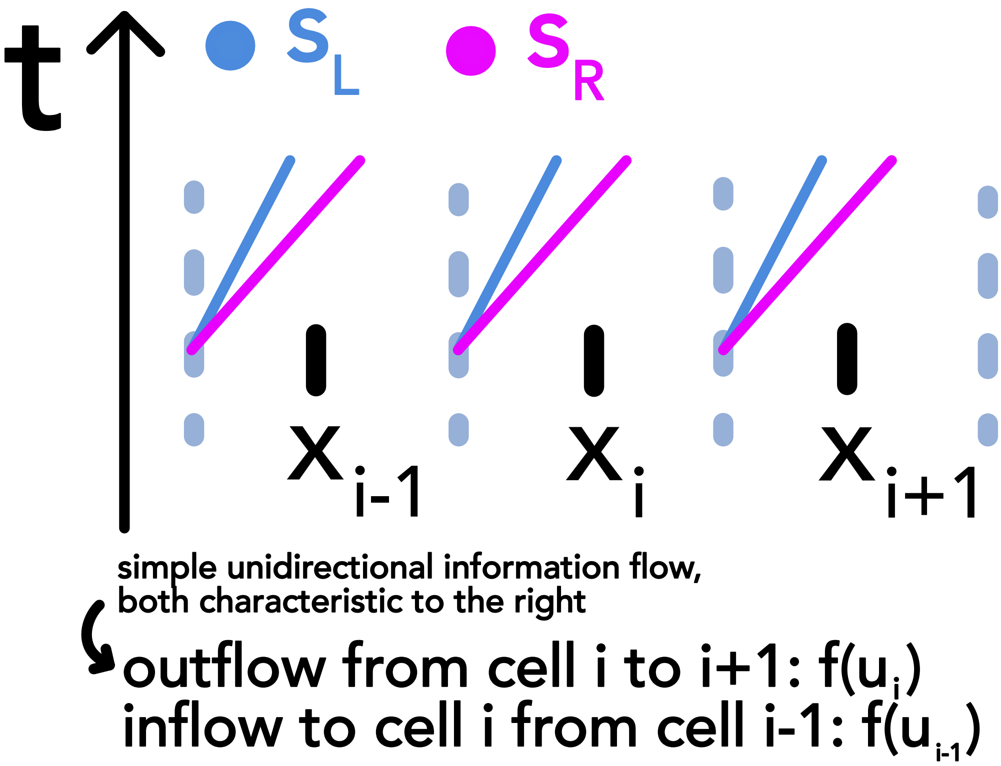


Figure 61: Godunov scheme with HLL solver for unidirectional information flow to the right.

We get

$$F^*_{i+\frac{1}{2}} = \begin{cases} F_i \text{ for } 0 < S_L \\ F^{HLL}_{i+\frac{1}{2}} \text{ for } S_L \leq 0 \leq S_R \\ F_{i+1} \text{ for } 0 > S_R \end{cases} , \quad F^*_{i-\frac{1}{2}} \text{ from } i \rightarrow i-1 \tag{278}$$

with

$$F^{HLL}_{i+\frac{1}{2}} = \frac{S_R F_{i+1} - S_L F_i + S_L S_R \left(U_{i+1} - U_i\right)}{S_R - S_L} \tag{279}$$

where we can combine this expression and the cases above to

$$F^{HLL}_{i+\frac{1}{2}} = \frac{S_R^+ F_{i+1} - S_L^- F_i + S_L^- S_R^+ \left(U_{i+1} - U_i\right)}{S_R^+ - S_L^-}, \quad S_R^+ = \max(0, S_R), \quad S_L^- = \min(0, S_L) \tag{280}$$

6.8.7 Pointers to extensions of the HLL scheme

- in HLLC an additional velocity between S_L and S_R is considered
- HLLD used for magnetohydrodynamics (MHD)

6.8.8 Ansätze for the maximum wave velocities S_L and S_R

Consider the gas velocity v as given by part of the state vector and sound speed c_s (given by the problem, here denoted a_L and a_R), then possible estimates are

- $S_L = v_L - a_L, S_R = v_R + a_R$
- $S_L = \min(v_L - a_L, v_R - a_R), S_R = \max(v_R + a_R, v_R + a_R)$
- Roe average, where we weigh dense areas as more important to the communication (leading to less smearing but instability) of information.

$$\begin{aligned} S_L &= \tilde{u} - \tilde{a} \\ S_R &= \tilde{u} + \tilde{a} \\ \tilde{u} &= \frac{\sqrt{\rho_L}u_L + \sqrt{\rho_R}u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\ \tilde{a} &= \left[(\gamma - 1)\left(\widetilde{H} - \frac{1}{2}\tilde{u}^2\right)\right]^{\frac{1}{2}} \text{ with the enthalpy} \\ H &= (e + P)/\rho \text{ and} \\ \widetilde{H} &= \frac{\sqrt{\rho_L}H_L + \sqrt{\rho_R}H_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \end{aligned} \tag{281}$$

6.9 Extension of Eulerian hydrodynamics to multiple dimensions

Based on our previous results, we can simulate the 1D conservation law

$$\partial_t \underline{U} + \partial_x \underline{F}(\underline{U}) = 0 \tag{282}$$

e.g. for an isothermal gas (so constant sound speed c_s) with

$$\underline{U} = \begin{pmatrix} \rho \\ \rho v_x \end{pmatrix}, \quad \underline{F} = \begin{pmatrix} \rho v_x \\ \rho v_x^2 + P \end{pmatrix}, \quad P = c_s^2 \rho \tag{283}$$

Let us formulate the Euler equations for a 3D fluid (see eq. 259) by seperating the flux by direction

$$\partial_t \underline{U} + \partial_x \underline{F} + \partial_y \underline{G} + \partial_z \underline{H} = 0 \tag{284}$$

with

$$\underline{F}: \text{ flux vector along } \underline{\hat{e}}_x, \quad \underline{G}: \text{ flux vector along } \underline{\hat{e}}_y, \quad \underline{H}: \text{ flux vector along } \underline{\hat{e}}_z$$

$$\underline{F} = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ \rho uw \\ u(\rho e + P) \end{pmatrix}, \quad \underline{G} = \begin{pmatrix} \rho v \\ puv \\ \rho v^2 + P \\ \rho vw \\ v(\rho e + P) \end{pmatrix}, \quad \underline{U} = \begin{pmatrix} \rho w \\ puw \\ \rho vw \\ \rho w^2 + P \\ w(\rho e + P) \end{pmatrix} \tag{285}$$

$$\text{state vector } \underline{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e \end{pmatrix}$$

and

$$\text{total specific energy per uni mass } e = e_{th} + \frac{1}{2}\left(u^2 + v^2 + w^2\right)$$
$$\text{pressure } P = (\gamma - 1)\rho e_{th}, \quad \text{thermal energy per unit mass } e_{th}$$
$$u: \text{ velocity in } \underline{\hat{e}}_x \text{ direction}, \quad v: \text{ velocity in } \underline{\hat{e}}_y \text{ direction}, \quad w: \text{ velocity in } \underline{\hat{e}}_z \text{ direction} \tag{286}$$

6.9.1 Dimensional splitting Ansatz

Idea: Seperately update dimensions (using our 1D solver) and combine.

From eq. 284 we make the following separation ansatz

$$\partial_t \underline{U} + \partial_x \underline{F} = \underline{0}, \quad \partial_t \underline{U} + \partial_y \underline{G} = \underline{0}, \quad \partial_t \underline{U} + \partial_z \underline{H} = \underline{0} \tag{287}$$

Note: While the state vector and fluxes are still $\in \mathbb{R}^5$ (the velocities in the other directions appear), we effectively have *augmented* 1D problems (the flux along x is not as directly coupled to w as it is to u)

To forward our state in 3D we have to sequence multiple augmented 1D steps (sweeps). For 2D this is illustrated in figure 62.

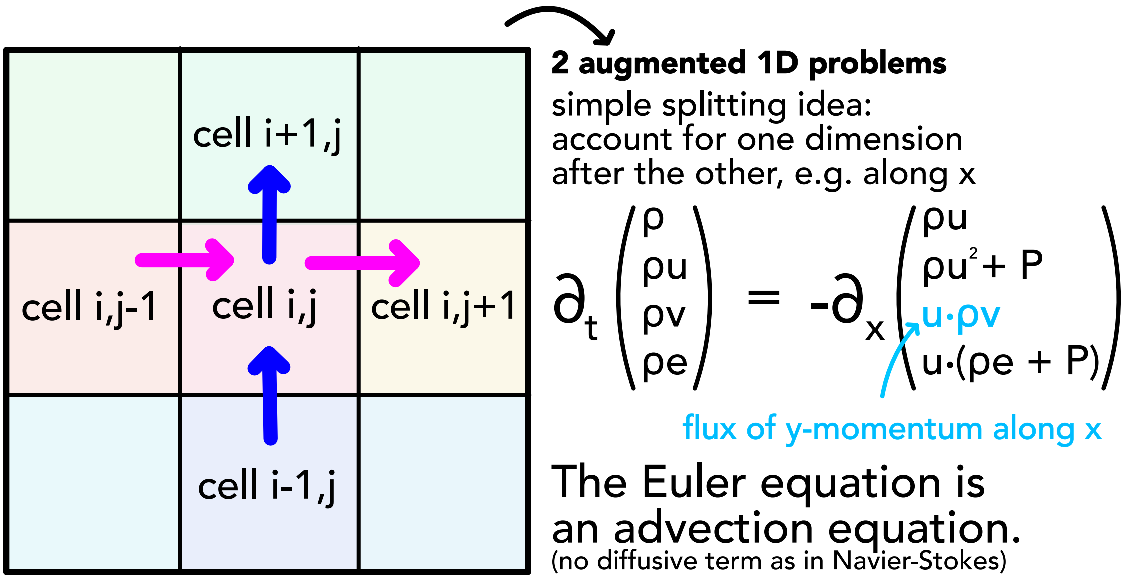


Figure 62: Splitting Ansatz for 2D

6.9.1.1 1st order ansatz

Assuming we already have a method to advance in one dimension then

$$\underline{U}^{(n+1)} = \mathcal{X}(\Delta t)\mathcal{Y}(\Delta t)\mathcal{Z}(\Delta t)\underline{U}^{(n)}, \quad \text{time evolution operators} \mathcal{X}, \mathcal{Y}, \mathcal{Z} \tag{288}$$

is a dimensionally split update scheme, which is exact for linear advection but not so for any higher order problem (first order reduction) as the steps in the dimensions are done separately.

6.9.1.2 2nd order accurate in 2D examples

$$\begin{aligned} \underline{U}^{(n+1)} &= \frac{1}{2}[X(\Delta t)\mathcal{Y}(\Delta t) + \mathcal{Y}(\Delta t)\mathcal{X}(\Delta t)]\underline{U}^{(n)} \\ \text{or } \underline{U}^{(n+1)} &= X\left(\frac{\Delta t}{2}\right)\mathcal{Y}(\Delta t)\mathcal{X}\left(\frac{\Delta t}{2}\right)\underline{U}^{(n)} \end{aligned} \tag{289}$$

6.9.2 2nd order accurate in 3D example

$$\underline{U}^{(n+1)} = x\left(\frac{\Delta t}{2}\right)\mathcal{Y}\left(\frac{\Delta t}{2}\right)z(\Delta t)\mathcal{Y}\left(\frac{\Delta t}{2}\right)\mathcal{X}\left(\frac{\Delta t}{2}\right)\underline{U}^{(n)} \tag{290}$$

where the 2nd order is based on the alternating reverse order application of the time evolution operators.

6.9.3 Unsplit schemes

Consider rectangular 2D cells. In a dimensionally split scheme we would make updates to a cell based on the information flow in only one direction and then the other based on the changed situation. In an unsplit scheme we apply both fluxes simultaneously.

In the case of rectangular cells in 2D, we have

$$\underline{U}_{i,j}^{(n+1)} = \underline{U}_{i,j}^{(n)} + \frac{\Delta t}{\Delta x}\left(\underline{F}_{i-\frac{1}{2},j} - \underline{F}_{i+\frac{1}{2},j}\right) + \frac{\Delta t}{\Delta y}\left(\underline{G}_{i,j-\frac{1}{2}} - \underline{G}_{i,j-\frac{1}{2}}\right) \tag{291}$$

In the situation of an unstructured mesh, we generally have

$$\underline{U}^{(n+1)} = \underline{U}^{(n)} - \frac{\Delta t}{V}\int \underline{F}\cdot d\underline{S} \text{ integral over cell surface, } d\underline{S} \text{ is the surface element vector, pointing outward} \tag{292}$$

which makes sense intuitively (the change coming from the borders distributes over the cells).

The situations are illustrated in figure 63.

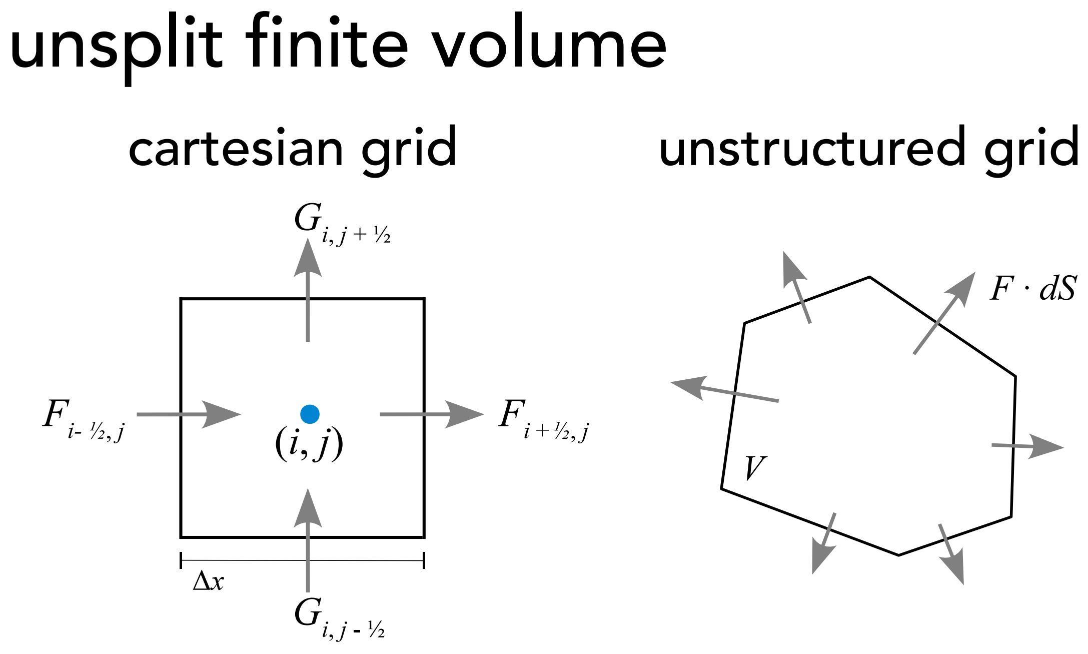


Figure 63: Unsplitting Ansatz Grids

6.10 Extensions for high-oder accuracy

6.10.1 What even is a schemes order?

Consider our numerical solution ρ_i sits on a grid of N points x_i, $i = 1, \dots, N$. Let $\rho(x)$ be the true solution. Then based on the mean L1 error

$$L1 = \frac{1}{N} \sum_{i=1}^{N} |\rho_i - \rho(x_i)| \tag{293}$$

we call a method

- first order accurate if $L1 \propto \Delta x \propto N^{-1}$ with $\Delta x = \frac{L}{N}$
- second order accurate if $L1 \propto \Delta x^2 \propto N^{-1}$
- ...

In the 2nd order accurate scheme, doubling the number of cells will quarter our error.

6.10.2 2nd order extension to Godunov's scheme by changing the reconstruction step from piecewise-constant to linear

Godunov's theorem states: Linear numerical schemes for solving partial differential equations (PDE's), having the property of not generating new extrema (monotone scheme), can be at most first-order accurate[a]. Bram van Leer (and Vladimir P. Kolgan[b]) first succeeded in **circumventing this** by nolinearly limiting the second order term as a function of the non-smoothness of the numerical solution (sustaining monotonicity) (a non-linear technique even for a linear equation, see flux limiters later on) (so we have (at least) 2nd order accuracy where the flow is smooth) (e.g. van Leer, 1979).

[a] In Godunov's scheme the dissipation is just strong enough to damp shorter waves before they get too much out of step and show up as oscillations on top of the larger features.

[b] Already in 1972, he developed a scheme 2nd order in space (using the (*crude*) minmod limiter) but only first order in time, using Forward Euler. Sadly, he succumbed to lung cancer in 1978, at the age of 37.

1. Estimate the fluid variables gradients, e.g. $\partial_x \rho$ (e.g. based on the averages on the neighboring cells)
2. Slope limit these gradients as otherwise we could introduce quite extreme values of the fluid variables at the cell boundaries, especially in case real fluid discontinuities are present

3. Estimate the values of the fluid variables at one interface by linear extrapolation

$$\rho^L_{i+\frac{1}{2}} = \rho_i + \frac{\Delta x}{2}(\partial_x \rho)_i, \quad \rho^R_{i+\frac{1}{2}} = \rho_{i+1} - \frac{\Delta x}{2}(\partial_x \rho)_{i+1} \tag{294}$$

See figure 64 for an illustration.

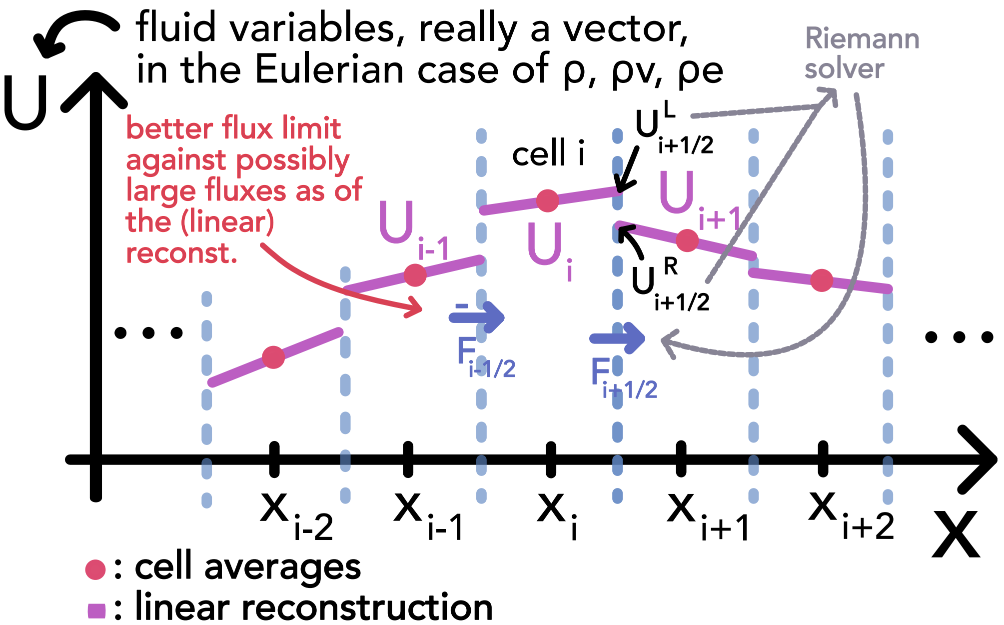


Figure 64: Linear extrapolation of the fluid variables to the cell interface.

4. Based on the extrapolated fluid variable values at the interface we apply our Riemann solver to find the flux and update the cell averages (although we do not have the piecewise-constant Riemann situation)

Problem: Above we have done a linear extrapolation to the boundary $\rho^L_{i+\frac{1}{2}}$ based on the gradient at the center $(\partial_x \rho)_i$. Note, however, that over our timestep Δt, ρ_i changes, so also our value at the boundary.

$$d\rho(x,t) = (\partial_x \rho)dx + (\partial_t \rho)dt \tag{295}$$

For more stability (and 2nd order accuracy), we do the extrapolation based on the gradient but also include the effect of $(\partial_t \rho)_i$ up intil half a timestep (as a proxy to the situation throughout the timestep).

$$\begin{gathered}\rho^L_{i+\frac{1}{2}} = \rho_i + (\partial_x \rho)_i \frac{\Delta x}{2} + (\partial_t \rho)_i \frac{\Delta t}{2}, \quad \rho^R_{i+\frac{1}{2}} = \rho_{i+1} - (\partial_x \rho)_{i+1} \frac{\Delta x}{2} + (\partial_t \rho)_{i+1} \frac{\Delta t}{2} \\ \rightarrow \text{ flux estimate } \underline{F}_{\text{Riemann}}\left(\underline{U}^L_{i+\frac{1}{2}}, \underline{U}^R_{i+\frac{1}{2}}\right) \text{ effectively at half-timestep}\end{gathered} \tag{296}$$

So for the entire state $\underline{U}$ we have

$$\underline{U}^L_{i+\frac{1}{2}} = \underline{U}_i + \left(\partial_x \underline{U}\right)_i \frac{\Delta x}{2} + \left(\partial_t \underline{U}\right)_i \frac{\Delta t}{2}, \quad \underline{U}^R_{i+\frac{1}{2}} = \underline{U}_{i+1} - \left(\partial_x \underline{U}\right)_{i+1} \frac{\Delta x}{2} + \left(\partial_t \underline{U}\right)_{i+1} \frac{\Delta t}{2}$$
$$\left(\partial_x \underline{U}\right)_i \text{ calculated from finite difference approach + slope limiting} \tag{297}$$

Idea of the MUSCL-Hancock scheme: Cell averages of the primitive fluid quantities are used to predict the values at the cell boundaries as $t + \frac{\Delta t}{2}$ and then use these prediction to calculate the fluxes and with them the primitive fluid quantities at $t + \Delta t$.

6.10.2.1 How to estimate the time derivatives $(\partial_t \underline{U})_i$? | MUSCL-Hancock scheme

Let us use the Euler equation in 1D (x is a scalar)

$$\partial_t \underline{U} + \partial_x \underline{F}(\underline{U}) \underbrace{=}_{\text{quasi-linear form}} \partial_t \underline{U} + \underline{\underline{J}}_{\underline{U}}(\underline{F}) \cdot \partial_x \underline{U} = 0 \rightarrow \boxed{\partial_t \underline{U} = -\underline{\underline{J}}_{\underline{u}}(\underline{F}) \cdot \partial_x \underline{U}}$$
$$\text{with Jacobian matrix } \underline{\underline{J}}_{\underline{U}}(\underline{F}) \text{ of } \underline{F}(\underline{U}) \text{ with respect to } \underline{U}, \quad \underline{\underline{J}}_{\underline{U}}(\underline{F})\Big|_{\underline{U}=\underline{U}} =: \underline{\underline{A}}(\underline{U}) \tag{298}$$

We can therefore estimate the derivative in time based on our estimation $\partial_x \underline{U}$ of the derivative in space, yielding the MUSCL-Hancock scheme (Monotonic Upstream Scheme for Conservation Laws)

$$\begin{aligned} \underline{U}^L_{i+\frac{1}{2}} &= \underline{U}_i + \left[\frac{\Delta x}{2}\underline{\underline{1}} - \underline{\underline{A}}(\underline{U})\frac{\Delta t}{2}\right] \left(\partial_x \underline{U}\right)_i \\ \underline{U}^R_{i+\frac{1}{2}} &= \underline{U}_{i+1} + \left[-\frac{\Delta x}{2}\underline{\underline{1}} - \underline{\underline{A}}(\underline{U})\frac{\Delta t}{2}\right] \left(\partial_x \underline{U}\right)_{i+1} \end{aligned} \tag{299}$$

- a 2nd order accurate extension of Godunov's scheme.

6.10.3 Idea and discussion of even higher order methods

We can

- use higher order polynomial reconstruction as in piecewise parabolic methods (also mind information transport by characteristic waves), see figure 65 (high order methods tend to create post shock oscillations → add dissipation mechanism / some flattening[9] to PPM)

[9] Which essentially means that locally we go back to lower order.

- even higher order polynomials are used in methods like ENO and WENO (find polynomials based on values from multiple cells (*larger stencil*))

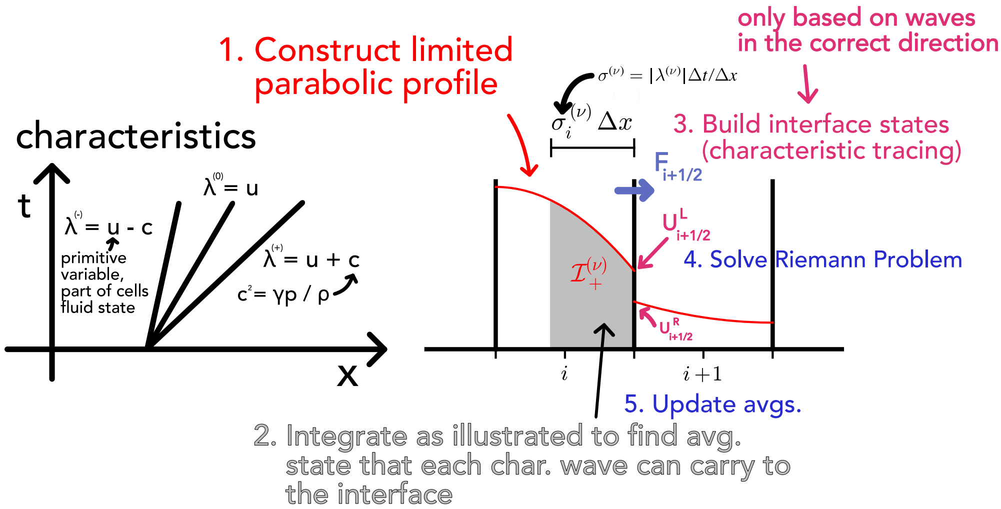


Figure 65: Piecewise parabolic reconstruction.

Note: Independent of the order, we only **store** one value (per variable) per cell in finite volume methods. In **finite element methods**, per cell polynomial representations of the state vector are stored (discontinuities are harder to capture though).

6.10.3.1 Discussion of higher order methods

Advantages and disadvantages of higher order methods can be found in table 11, those of lower order methods in table 12.

Pro higher	Con higher
• sharp solution • more accurate (sharper solutions alone can be inaccurate, e.g. when the bulk position is inaccurate)	• more expensive • strong oscillations at discontinuities (principle illustrated in figure 66) • crash at high Mach numbers

Table 11: Advantages and disadvantages of higher order methods.

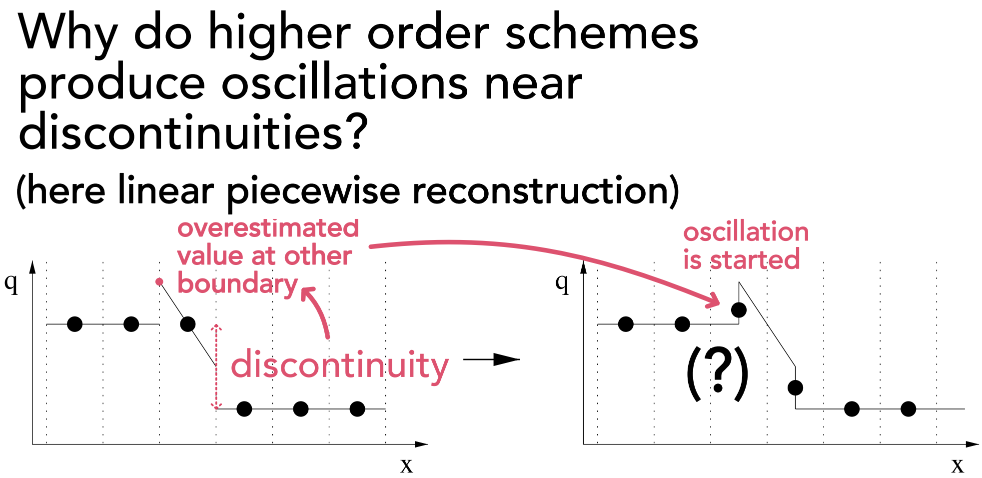


Figure 66: Illustration of the principle of oscillations at discontinuities in higher order methods.

Pro lower	Con lower
• stable for complex flows • no ocillations at discontinuities • do not crash at high Mach numbers	• smear out solutions • slower convergence to accurate solutions • less accurate

Table 12: Advantages and disadvantages of lower order methods.

6.11 Flux / slope limiters | adaptively switching between a high and low order method

Problem: A system can contain very boring and strongly dynamic parts. It is very difficult to choose which solver is best for the global problem for all timesteps.

Idea: Dynamically switch between orders and solvers. Use 2nd order where possible and 1st order where necessary (e.g. for discontinuities).

Let us for simplicity consider a 1D problem with a single state variable u, e.g. the viscous Burgers equation

$$\partial_t u + \partial_x \left(\frac{1}{2} u^2 - \nu \partial_x u \right) = 0 \tag{300}$$

(which can be analytically solved using the Cole-Hopf transform $u = -2\nu \frac{1}{\phi} \partial_x \phi$[10]).

[10] This is nice if we want to test different flux limiters against a ground-truth result.

Let $\mathcal{F}^H_{i+\frac{1}{2}}$ be a high-order flux computation and $\mathcal{F}^L_{i+\frac{1}{2}}$ be a low-order flux computation. We use the flux

$$
\begin{gathered}
F^{(n)}_{i+\frac{1}{2}} = \mathcal{F}^{L,(n)}_{i+\frac{1}{2}}(U_i, U_{i+1}) + \phi^{(n)}_{i+\frac{1}{2}}(r) \cdot \left(\mathcal{F}^{H,(n)}_{i+\frac{1}{2}}(U_i, U_{i+1}) - \mathcal{F}^{L,(n)}_{i+\frac{1}{2}}(U_i, U_{i+1})\right) \\
\text{high order for } \phi^{(n)}_{i+\frac{1}{2}}(r) = 1, \quad \text{low order for } \phi^{(n)}_{i+\frac{1}{2}}(r) = 0
\end{gathered}
\tag{301}
$$

with

$$
\begin{gathered}
\text{flux limiter } \phi^{(n)}_{i+\frac{1}{2}}(r) \text{ based on the ratio } r = \frac{U_i - U_{i-1}}{U_{i+1} - U_i} \\
\text{large if the jump of interest between } U_i \text{ and } U_{i+1} \text{ is large compared to the jump between } U_{i-1} \text{ and } U_i
\end{gathered}
\tag{302}
$$

with and exemplary flux limiter being

$$
\phi_{\text{minmod}} = \max(0, \min(1, r)), \quad r > 1 \to \phi_{\text{minmod}} = 1 \to \text{high order} \tag{303}
$$

(illustrated in figure 67) where it makes sense that for a big r we use the higher order method: If the jump between i and $i+1$ is small compared to the jump between $i-1$ and i, we can assume that the solution is smooth and use the higher order method (no dicontinuity expected).

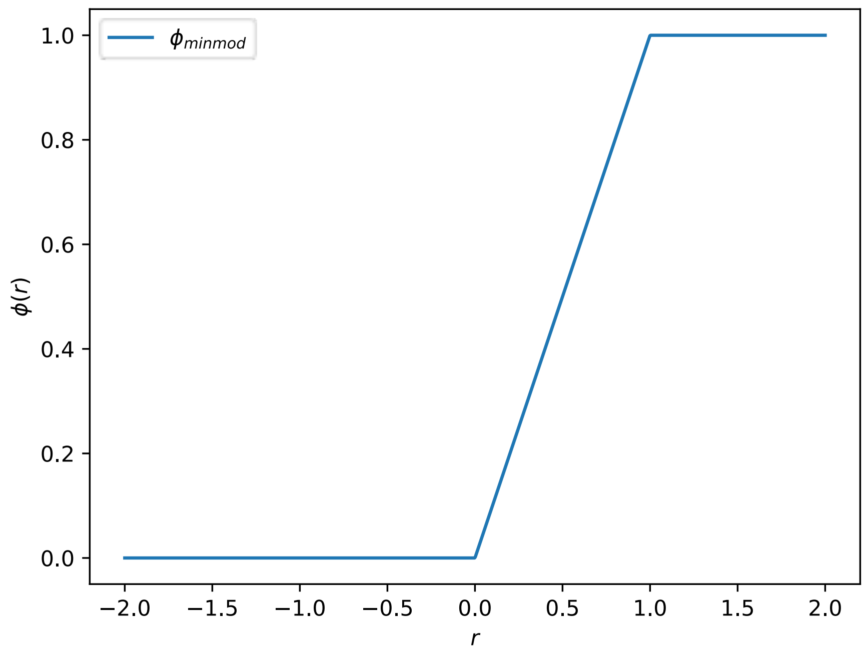


Figure 67: Minmod flux limiter.

Why is ϕ called a flux limiter? Higher order schemes can have larger slopes and predict larger fluxes, taking the lower order will has less of this problem.

6.11.1 Possibly advantageous properties of the flux limiter

It is often advantageous for the limiters to be symmetric with the property

$$\frac{\phi(r)}{r} = \phi\left(\frac{1}{r}\right) \tag{304}$$

so that the switch works the same for forward and backward facing gradients (mind the definition of r) (?).

Another property is based on the total variation

$$\mathrm{TV}(\underline{U}) = \sum_i |U_{i+1} - U_i| \tag{305}$$

(where here $\underline{U}$ is the vector over the grid points not fluid variables), called total variation diminishing (TVD) property

$$\mathrm{TV}(\underline{U}^{(n+1)}) \leq \mathrm{TV}(\underline{U}^{(n)}) \tag{306}$$

which means no oscillation will appear but real ocscillations in the system will also not grow but rather smeared out. Minmod is TVD.

7 Smoothed Particle Hydrodynamics - Lagrangian Particle Method

Idea: In Smoothed Particle Hydrodynamics (SPH) we approximately solve the fluid equations numerically by replacing the fluid with a set of particles, we call *SPH-particles*[a]. The equations of motion and properties of those particles are followed from the Lagrangian form of the continuity equations for the fluid. We then forward these particles in time using e.g. the leapfrog or semi-implicit Euler scheme.

[a]Characterized by their position and velocity. Additionally, hydrodynamic variables, e.g. ρ_i, T_i, are derived at the particles positions $\underline{x}_i$. Not to be confused with the real particles making up the fluid.

From this it makes sense that the main ingredients to the baseline scheme will be

- formulate the fluid equations in their Lagrangian[11] form
- formulate an algorithm to update the SPH-particles positions and velocities based on the Lagrangian fluid equations
- find expressions for the quantities used in the update-steps which goes hand in hand with finding how to get from the SPH-particle-perspective to continuous fluid quantities

Such a mesh-free scheme has some key advantages

- This particle representation of SPH has great conservation features - energy, linear momentum, angular momentum mass and specific thermodynamic entropy (more later; if we do not add artificial viscosity) are conserved (as we follow particles there is no loss of mass etc.).
- No advection errors, scheme is fully Galilean invariant (unlike mesh-based Eulerian techniques)
- As of the Lagrangian character, the local resolution of SPH follows the mass flow automatically → adaptive resolution, good for problems with vastly different densities

[11]Co-moving with the flow rather than fixed as in the Eulerian perspective.

7.1 Lagrangian fluid equations (i.e. as material derivatives)

7.1.1 Continuity equation

Written as a material derivative, the continuity equation is

$$D_t\rho = -\rho(\underline{\nabla} \cdot \underline{v}), \quad \text{fluid mass density } \rho, \quad \text{fluid velocity } \underline{v} \tag{307}$$

which is zero for an incompressible fluid.

7.1.2 Navier-Stokes equation | Conservation law of Linear Momentum

A natural extension of Newtons 2nd law to continua is (including internal and external forces)

$$\begin{gathered} \rho D_t \underline{v} = \sum \text{forces} = -\underline{\nabla} P + \underline{\nabla} \cdot \underline{\underline{\Pi}} + \rho \underline{g} \\ \text{stress tensor } \underline{\underline{\Pi}}, \quad \text{pressure } P, \quad \text{external accelerations } \underline{g} \end{gathered} \tag{308}$$

where a general approach to $\underline{\underline{\Pi}}$ is the Cauchy-Stress tensor for compressible flow

$$\begin{gathered} \underline{\underline{\Pi}} = \left\{ \mu \left[\underline{\nabla}\underline{v}^T + (\underline{\nabla}\underline{v}^T)^T - \frac{2}{3}(\underline{\nabla} \cdot \underline{v})\underline{\underline{1}} \right] + \zeta(\underline{\nabla} \cdot \underline{v})\underline{\underline{1}} \right\} \\ \text{shear viscosity } \mu, \quad \text{bulk viscosity } \zeta \end{gathered} \tag{309}$$

where for incompressible flow one yields

$$\rho D_t \underline{v} = -\underline{\nabla} P + \mu \underline{\nabla}^2 \underline{v} + \rho \underline{g} \tag{310}$$

Note: In an incompressible setting, the pressure P can be interpreted as a Lagrange multiplier which has to be chosen such that incompressibility really holds. Otherwise P is determined by a state equation, e.g. very basic $P(\rho) = k\left(\frac{\rho}{\rho_0} - 1\right), k > 0$ (variation of the ideal gas equation) or isothermal $P(\rho) = P_0 + c_0^2(\rho_0 - \rho)$, as previously discussed.

7.1.3 Energy equation

While me can forward the system only based on the Navier-Stokes equation and closure, let us write down the energy equation.

Let ϵ be the energy per volume so the energy per mass is $e = \frac{\epsilon}{\rho}$.

From basic thermodynamics, we can write for the internal energy U

$$
\begin{gathered}
dU = dQ + dW = dQ - PdV = TdS - PdV, \quad U = \epsilon V = eM \\
\text{volume } V = \frac{M}{\rho} \to dV = -\frac{M}{\rho^2} d\rho, \quad \text{total mass } M, \quad \text{entropy } S, \quad \text{pressure } P
\end{gathered} \tag{311}
$$

using this we find

$$
\begin{aligned}
\frac{de}{dt} &= \partial_t e + (\underline{v} \cdot \underline{\nabla})e = \frac{1}{M}\frac{dU}{dt} = \frac{1}{M} T \frac{dS}{dt} - \frac{1}{M} P \frac{dV}{dt} \\
&= T\frac{ds}{dt} + \frac{P}{\rho^2}\frac{d\rho}{dt} \underbrace{=}_{\text{continuity eq.}} T\frac{ds}{dt} - \frac{P}{\rho}(\underline{\nabla} \cdot \underline{v})
\end{aligned} \tag{312}
$$

7.2 A simple SPH fluid simulator*

Based on the Navier-Stokes equation in Lagrangian form we can construct a simple fluid simulator, here using semi-implicit aka symplectic Euler[12]. Pressure and viscosity forces are calculated separately. We still have to mind the CFL criterion (more on this later).

To code up our simulator we have to answer

- How do we construct the density from the SPH-particles positions $\underline{x}_i$?
- How do we calculate the gradients over fluid variables, so $\underline{\nabla} P$?

[12] The PySPH package for instance uses a second order predictor-corrector method.

```
for sph_particle_i in sph_particles:
    # reconstruct density $\rho_i$ at $\underline{x}_i$
for sph_particle_i in sph_particles:
    ### viscous-force calculation in case of a viscous fluid
    # / to make shocks resolvable using artificial viscosity
    # compute $\underline{a}_i^{viscosity}$ in the incomp. case $= \nu\underline{\nabla}^2\underline{v}_i$
    # $\underline{v}_i^* = \underline{v}_i + \Delta t(\underline{a}_i^{viscosity} + \underline{g})$, external accelerations $\underline{g}$

    ### pressure force calculation
    # compute $\underline{a}_i^{pressure} = -\frac{1}{\rho_i}\underline{\nabla}P$

    ### forward the particles, here using symplectic Euler
    # $\underline{v}_i(t+\Delta t) = \underline{v}_i^* + \Delta t \cdot \underline{a}_i^{pressure}$
    # $\underline{x}_i(t+\Delta t) = \underline{x}_i + \Delta t \cdot \underline{v}_i(t+\Delta t)$
```

Code-Snippet 1: Simple SPH fluid simulator. We need two loops, as to calculate e.g. the pressure force on one SPH-particle, the densities at positions of other SPH-particles are necessary.

- How do we handle viscosity?

7.3 Smooth then discretize - smoothing kernels and their usage

Fluid quantities like the density are estimated though a kernel summation interpolant. Start by replacing a general fluid quantity $F(\underline{r})$ with a smoothed version by convoluting it with a smoothing kernel

$$\begin{aligned} &F(\underline{r}) \to F_S(\underline{r}) \equiv \langle F(\underline{r})\rangle = \int F\left(\underline{r}'\right) W\left(\underline{r}-\underline{r}', h\right) d^3\underline{r}', \quad \text{smoothing width } h \\ &\text{kernel } W \text{ with } \int W\left(\underline{r}', h\right) d^3\underline{r}' = 1, \quad \langle F(\underline{r})\rangle \underset{h\to 0}{\longrightarrow} F(\underline{r}), \text{ i.e. } W(\underline{r}, h) \underset{h\to 0}{\longrightarrow} \delta(\underline{r}) \end{aligned} \tag{313}$$

where the kernel has to be normed to not modify e.g. the total mass, and also differentiable (so that our fluid quantities are smooth). Typically, a spherical kernel is used

$$W(\underline{r}, h) = W(r, h) \tag{314}$$

which could be a Gaussian - but a Kernel with finite support, for instance a cubic spline, is better (more on that later).

7.3.1 Properties of the smoothing | approach for calculating derivatives of the smoothed fluid quantities

The smoothed version is 2nd order accurate with respect to the smoothing length (no first order correction as of the symmetry of the kernel)

$$\langle F(\underline{r})\rangle = F(\underline{r}) + \mathcal{O}\left(h^2\right) \tag{315}$$

We can also find

$$\begin{aligned} \langle F(\underline{r}) + G(\underline{r})\rangle &= \langle F(\underline{r})\rangle + \langle G(\underline{r})\rangle \\ \langle F(\underline{r}) \cdot G(\underline{r})\rangle &= \langle F(\underline{r})\rangle \cdot \langle G(\underline{r})\rangle + \mathcal{O}\left(h^2\right) \\ &\boxed{\frac{d}{dt}\langle F(\underline{r})\rangle = \left\langle \frac{dF(\underline{r})}{dt} \right\rangle} \\ \underline{\nabla}\langle F(\underline{r})\rangle \quad &\underbrace{=}_{\text{Kernel with compact support}} \quad \langle \underline{\nabla} F(\underline{r})\rangle \end{aligned} \tag{316}$$

where the main result that will allow us to make the fluid equation algebraic is

$$\boxed{\langle \underline{\nabla} F(\underline{r})\rangle = \underline{\nabla}\langle F(\underline{r})\rangle = \int F\left(\underline{r}'\right) \underline{\nabla} W\left(\left|\underline{r}-\underline{r}'\right|, h\right) d^3\underline{r}'} \tag{317}$$

so we weight $F(\underline{r}')$ using the gradient of the kernel which can be pre-computed.

7.3.2 Discrete formulation of the smoothing

We now introduce the SPH-particles at positions $\underline{r}_i$ where the fluid variable has value $F_i = F(\underline{r}_i)$. We assign a mass m_i to those particles. Together with the density $\rho_i = \rho(\underline{r}_i)$ we can write

$$\Delta r_i^3 \sim \frac{m_i}{\rho_i} \tag{318}$$

With this, assuming that those SPH-particles densely sample the space of interest, we can write the smoothed fluid quantity as

$$\boxed{F_s(\underline{r}) \equiv \langle F(\underline{r})\rangle \simeq \sum_{j=1}^{N_i} \frac{m_j}{\rho_j} F_j W\left(\underline{r}-\underline{r}_j, h\right), \quad \text{number of neighbors } N_i} \tag{319}$$

Note: While this is similar to Monte-Carlo integration, the evaluation points are our SPH-particles which here turns out to be favorable over random sampling, as the distances between the particles tend to equilibrate due to pressure forces (making the interpolation smaller - still resulting noise is a problem of SPH).

$F_s(\underline{r})$ is defined everywhere and differentiable as the kernel is differentiable.

For the density we can write

$$\boxed{\rho_s(\underline{r}) = \sum_{j=1}^{N_i} m_j W(\underline{r}-\underline{r}_j, h)} \tag{320}$$

The smoothing length h should at least be larger than the particle distance.

7.3.3 Why a kernel with compact support is preferred?

Consider we are interested in the density ρ_i at $\underline{x}_i$. Note that the density at any point is based on the overlap of multiple smoothing kernels (see eq. 320).

Consider a Gaussian kernel

$$W_{\text{Gaussian}}(||\underline{r}_i - \underline{r}_j||, h) = \frac{1}{(\pi h^2)^{\frac{d}{2}}} \exp\left(-q^2\right), \quad q := \frac{||\underline{r}_i - \underline{r}_j||}{h}, \quad \text{dimension } d \tag{321}$$

then as of the infinite support, for the density ρ_i we would have to sum over all particles, and the density calculation at all SPH-particles $i = 1, \ldots, N$ would be $\mathcal{O}(N^2)$.

Now for a cubic spline Kernel

$$\begin{aligned} W(q) &= \sigma_3 \left[1 - \frac{3}{2}q^2\left(1 - \frac{q}{2}\right)\right], && \text{for } 0 \leq q \leq 1 \\ &= \frac{\sigma_3}{4}(2-q)^3, && \text{for } 1 < q \leq 2 \\ &= 0, && \text{for } q > 2 \end{aligned} \tag{322}$$

with normalization

$$d = 1 : \sigma_3 = \frac{2}{3h}, \quad d = 2 : \sigma_3 = \frac{10}{7\pi h^2}, \quad d = 3 : \sigma_3 = \frac{1}{\pi h^3} \tag{323}$$

the density calculation is only $\mathcal{O}(N_{ngb}N)$ with N_{ngb} being the average number of neighbors considered depending on the choice of h.

Note: The support size of SPH-particles is usually chosen to be $2h$ so that $W_{\text{Gaussian}}(||\underline{r}_i - \underline{r}'||, h) = 0$ for $||\underline{r}_i - \underline{r}'|| > 2h$, see figure 68.

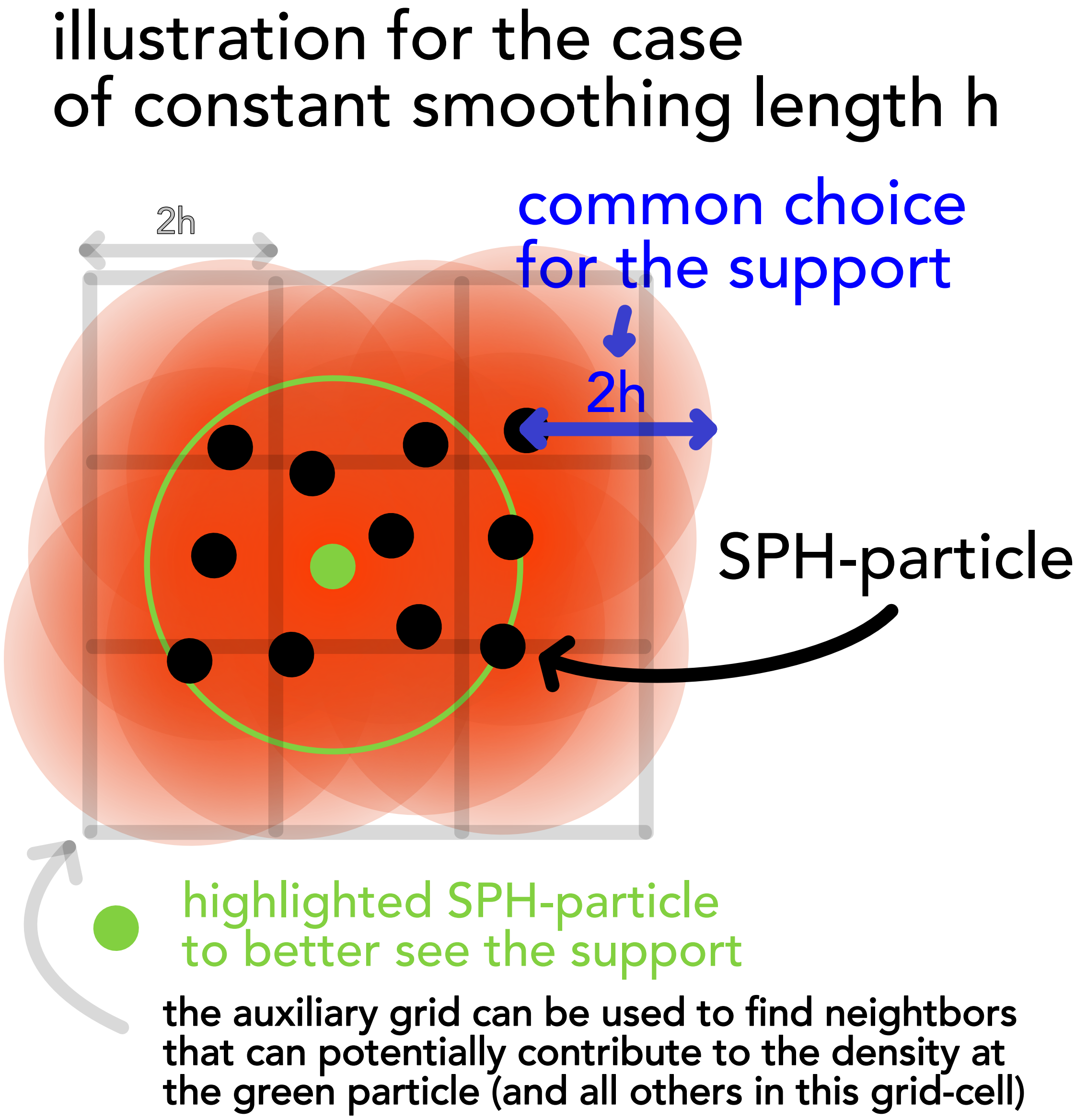


Figure 68: SPH illustration for fixed h.

7.3.4 How to make the smoothing length h variable in space to account for variations in the density? | sampling procedure in SPH - scatter and gather approach

Problem: Choosing a good overall smoothing length h is a problem in any algorithm using kernels (for instance also in Kernel Density Estimation). If h is chosen too large, we loose the details of the density distribution, if h is chosen to small, we get not a smooth density distribution but one with peaks at the positions of our SPH-particles. It makes more sense to use an adaptive smoothing length which is smaller in regions where SPH-particles are denser (to still resolve details there) and larger where they are more rarefied, so that in such regions the overall density would still make sense for a - in reality - continuous fluid, see figure 69

.

Idea: Use a variable smoothing length h.

There are two general approaches for introducing a variable smoothing length ρ_i into the calculation of our fluid quantities - the scatter and gather approach, as shown in table 13 and illustrated in figure 70.

Symmetry problem: Consider the effect of a variable smoothing length on how SPH-particles affect each other. In the schemes above this is not necessarily symmetric, leading to a force asymmetry and Newton's third law being broken (no conservation of total angular momentum).

We therefore have to symmetrize the equations of motion in $h_i = h(\underline{r}_i)$ and $h_j = h(\underline{r}_j)$. We make forces antisymmetric by substituting e.g.

$$h_{ij} = \frac{h_i + h_j}{2}, \quad \text{or geometric mean} \quad h_{ij} = \sqrt{h_i h_j} \tag{326}$$

for h_i and h_j in the force calculations. Therefore, a symmetric approach to the density is

$$\rho_s\left(\underline{r}_i\right) = \sum_{j=1}^{N_i} m_j W\left(r_{ij}, h_{ij}\right), r_{ij} = \left|\underline{r}_i - \underline{r}_j\right| \tag{327}$$

Note: As h is a function of $\underline{r}$ in the gather scheme, we must account for this in

$$\underline{\nabla} W\left(\left|\underline{r} - \underline{r}'\right|, h\right) = \underline{\nabla} W\left(\left|\underline{r} - \underline{r}'\right|, h\right)\Big|_{h=const.} + (\underline{\nabla} h)\partial_h W\left(\left|\underline{r} - \underline{r}'\right|, h\right)\Big|_{h=\text{ const.}} \tag{328}$$

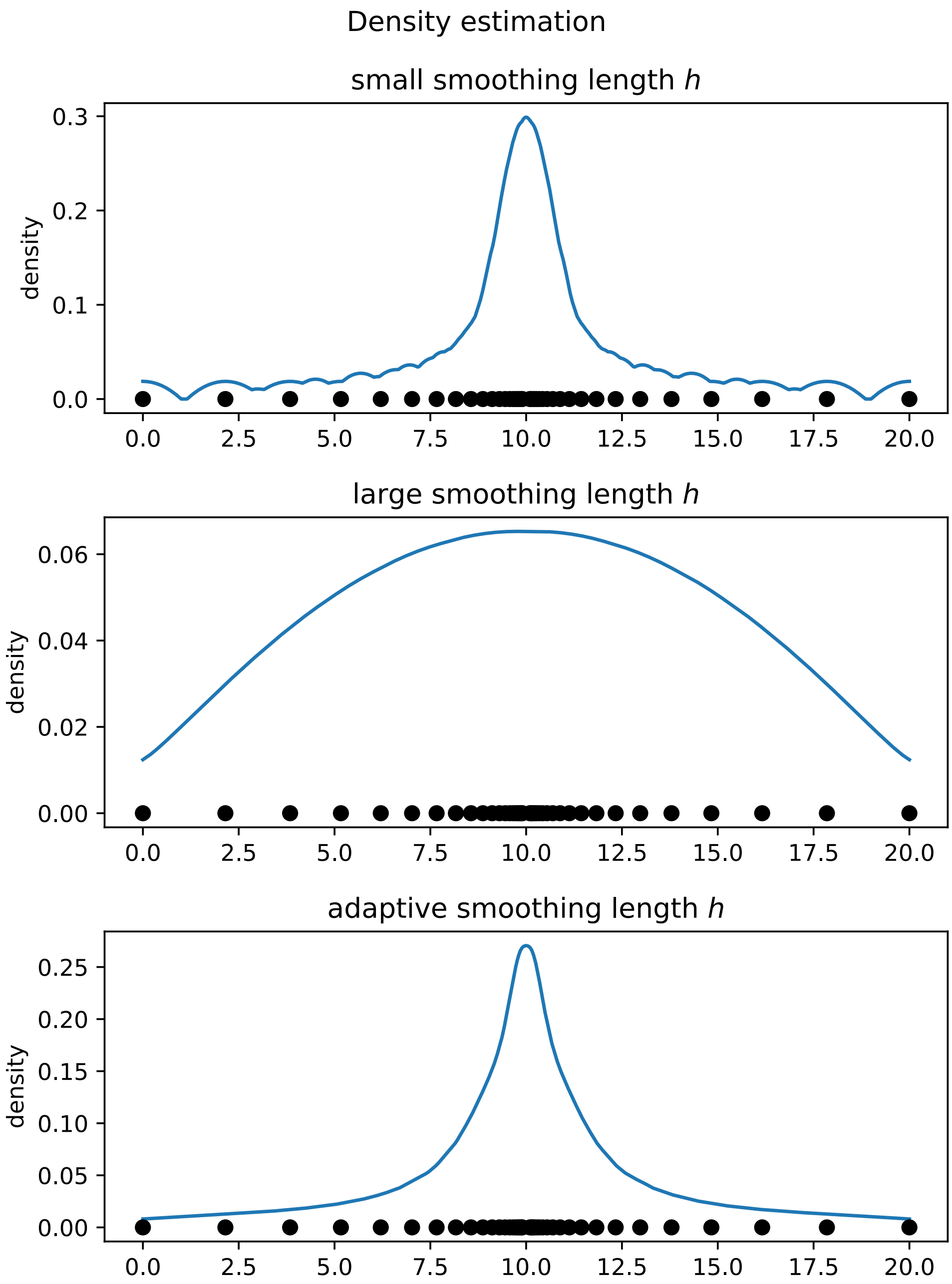


Figure 69: Density estimation

7.3.4.1 How to choose h_i?

We adjust h_i so that we always consider $50 \lesssim 50 \lesssim 500$ neighbors.

One choice is to choose h to keep N_{ngb} constant. Another is to scale h according to the density (higher for lower densities), e.g.

$$h_i = h_0 \left(\frac{\rho_0}{\rho_i}\right)^3, \quad \frac{dh}{dt} = -\frac{1}{3}\frac{h}{\rho}\frac{d\rho}{dt} \underbrace{=}_{\text{continuity eq.}} \frac{1}{3} h \underline{\nabla} \cdot \underline{v}, \quad \text{dimension } d \tag{329}$$

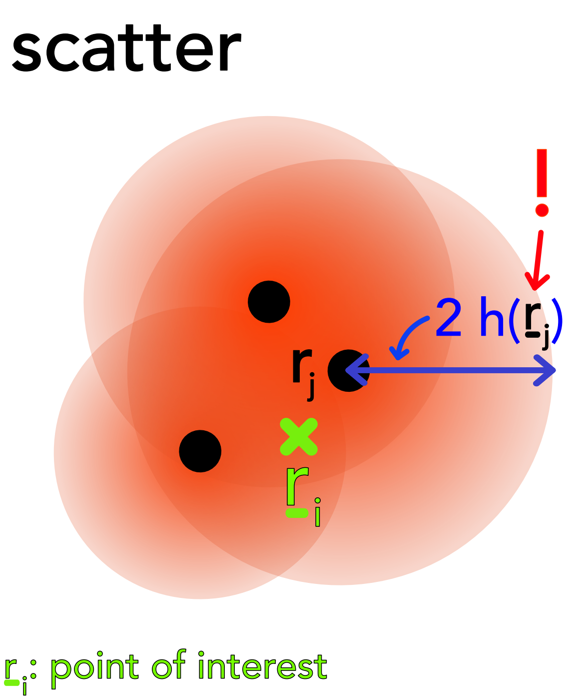
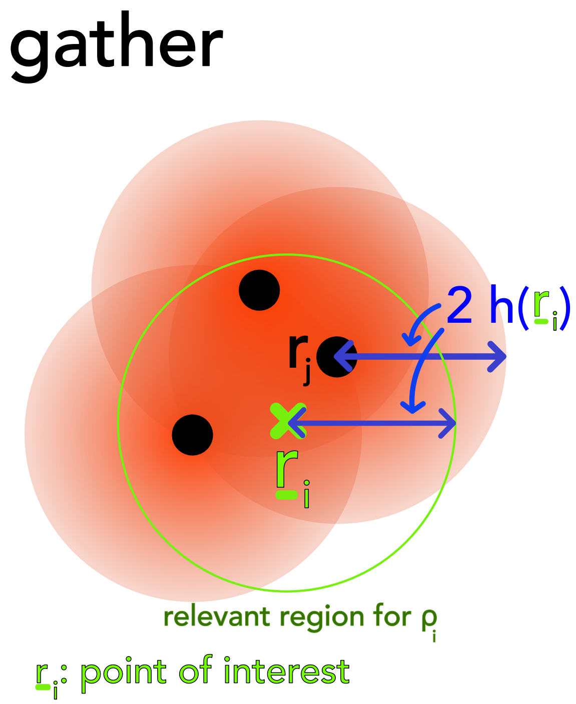
scatter each SPH-particle is assigned a smoothing length, the density at any point is calculated from the overlap of all surrounding density distributions	**gather** The observer at a position $\underline{r}$ has it's smoothing length h and assigns it to all SPH-particles in the relevant region. Assuming kernels with a compact support $2h$ this relevant region - starting from the observer - has radius $2h$.
	
$\langle F(\underline{r})\rangle = \int_{N_i} F\left(\underline{r}'\right) W\left(\underline{r}-\underline{r}', h\left(\underline{r}'\right)\right) d^3\underline{r}'$ $\rho_s\left(\underline{r}_i\right) = \sum_{j=1}^{N_i} m_j W\left(r_{ij}, h_j\right), r_{ij} = \left\lvert \underline{r}_i - \underline{r}_j \right\rvert$ (324)	$\langle F(\underline{r})\rangle = \int_{N_i} F\left(\underline{r}'\right) W\left(\underline{r}-\underline{r}', h(\underline{r})\right) d^3\underline{r}'$ $\rho_s\left(\underline{r}_i\right) = \sum_{j=1}^{N_i} m_j W\left(r_{ij}, h_i\right), r_{ij} = \left\lvert \underline{r}_i - \underline{r}_j \right\rvert$ (325)
The total mass is conserved in the scatter approach, $\int \rho_s d^3\underline{r} = \sum m_i$	The total mass is not conserved in the gather approach, the error is only $\mathcal{O}(h^2)$ thoug.

Table 13: Scatter and gather approach for introducing a variable smoothing length h.

so that the mass ρh^3 is constant.

The neighbors can be found using a tree structure for partitioning space.

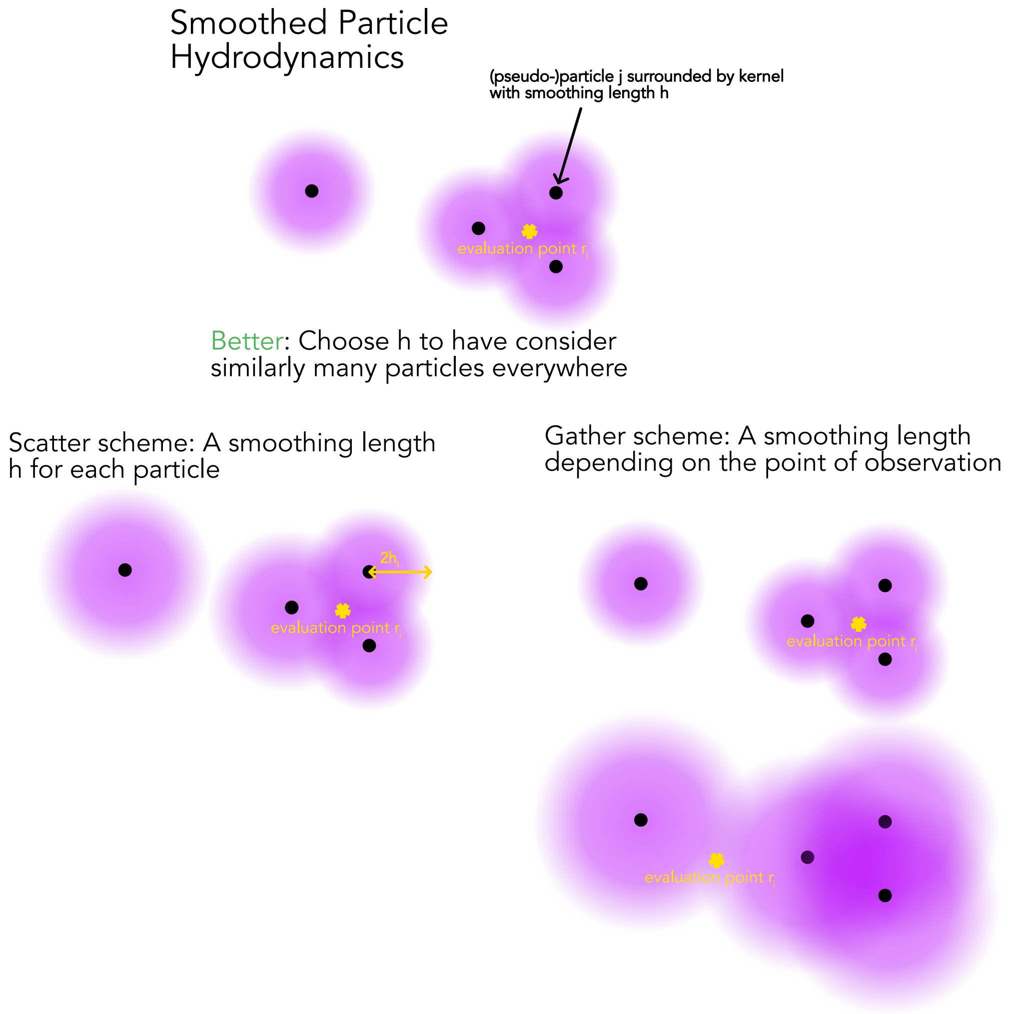


Figure 70: Scatter and gather approach for introducing a variable smoothing length h.

7.4 SPH continuity equation and equations of motion

We now know how to construct the densities ρ_i at the SPH-particles positions in one loop. Those densities are required for subsequent calculation of pressure forces etc. in a second loop.

7.4.1 SPH continuity equation

While better not used in forwarding the system, the SPH continuity equation can still be of interest.

Directly applying eq. 317 to the continuity equation in Lagrangian form, so $D_t\rho = -\rho\underline{\nabla} \cdot \underline{v}$, turns out to be extremely sensitive to the particle distribution, we can rewrite

$$\rho\underline{\nabla} \cdot \underline{v} = \underline{\nabla} \cdot (\rho\underline{v}) - \underline{v} \cdot \underline{\nabla}\rho \tag{330}$$

and applying eq. 317 and 319 we find for the i-th SPH-particle

$$\langle \underline{\nabla}\underline{v}\rangle_i = \frac{1}{\rho_i}\left[\langle \underline{\nabla}_i \cdot (\rho\underline{v})_i\rangle - \underline{v}_i \cdot \langle \underline{\nabla}\rho\rangle_i\right] = \frac{1}{\rho_i}\sum_{j=1}^{N_i} m_j \left(\underline{v}_j - \underline{v}_i\right)\underline{\nabla}_i W\left(r_{ij}, h_{ij}\right) \tag{331}$$

(which has the advantage that as it should be for all equal velocities $\underline{v}_i = \underline{v}_j$ we have $\langle \underline{\nabla}\underline{v}\rangle_i = 0$).

and thus the SPH continuity equation

$$\frac{d\rho_i}{dt} \simeq \sum_{j=1}^{N_i} m_j \underline{v}_{ij}\underline{\nabla}_i W\left(r_{ij}, h_{ij}\right), \quad \underline{v}_{ij} = \underline{v}_i - \underline{v}_j \tag{332}$$

7.4.2 Gradients in SPH

Note: There are multiple ways to obtain the SPH equations of motion. One is to find an expression for gradients and apply it to the Euler equations, another a variational approach, which directly guarantees certain conservation laws[a]

[a] There we start from the Lagrangian for inviscid (zero-viscosity) flow and then derive the Lagrangian equations of motion. The symmetries of the Lagrangian and absence of explicit time dependence directly give us energy conservation, momentum conservation from the translational invariance and momentum conservation from the rotational invariance.

As with the density, it is better not to directly apply eq. 317 but better to use the identities

$$\begin{aligned}\underline{\nabla} \cdot F(\underline{r}) &= \frac{1}{\rho}[\underline{\nabla} \cdot (\rho F(\underline{r})) - F(\underline{r})\underline{\nabla} \cdot \rho] \\ \underline{\nabla} \cdot F(\underline{r}) &= \rho\left[\underline{\nabla} \cdot \left(\frac{F(\underline{r})}{\rho}\right) + \frac{F(\underline{r})}{\rho^2}\underline{\nabla} \cdot \rho\right]\end{aligned} \tag{333}$$

to obtain

$$\langle \underline{\nabla} \cdot F(\underline{r})\rangle_i = \frac{1}{\rho_i}\sum_{j=1}^{N_i} m_j \left[F\left(\underline{r}_j\right) - F\left(\underline{r}_i\right)\right] \cdot \underline{\nabla}_i W_{ij}, \quad W_{ij} = W\left(r_{ij}, h_{ij}\right) \tag{334}$$

and the symmetric form

$$\langle \underline{\nabla} \cdot F(\underline{r})\rangle_i = \rho_i \left[\sum_{j=1}^{N_i} m_j \left[\frac{F\left(\underline{r}_j\right)}{\rho_j^2} + \frac{F\left(\underline{r}_i\right)}{\rho_i^2}\right] \cdot \underline{\nabla}_i W_{ij}\right] \tag{335}$$

7.4.3 SPH Euler equation | The central ingredient to making our simple fluid simulator work

With the gradient-expression, we can tackle the Euler equation in convective (Lagrangian) from (Navier-Stokes without stress tensor / external forces), in other words we can find the **acceleration as of pressure** $\underline{a}_i^{\text{pressure}}$.

$$\underline{a}^{\text{pressure}} = -\frac{\underline{\nabla} P}{\rho} \tag{336}$$

so in the (anti)symmetric form

$$\boxed{\underline{a}_i^{\text{pressure}} = -\sum_{j=1}^{N_i} m_j \left[\frac{P_j}{\rho_j^2} + \frac{P_i}{\rho_i^2}\right] \cdot \underline{\nabla}_i W_{ij}} \tag{337}$$

This is the acceleration we need to get our simple fluid simulator working - P follows from the equation of state.

Note: Artificial viscosity will have to be added to allow for the treatment of shocks.

The equation is antisymmetric in i and j so momentum is conserved locally and globally (also follows from the variational method).

Other symmetrizations

$$\text{e.g. } \frac{1}{2}\left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right) \quad \leftrightarrow \quad \frac{\sqrt{P_i P_j}}{\rho_i \rho_j} \tag{338}$$

and additional correction factors are possible.

7.4.4 Including further accelerations

Further accelerations like self gravity can be calculated as usual

$$\underline{g}_i = -\underline{\nabla}\Phi_i = -G\sum_{j=1}^{N_i} m_j \frac{\underline{r}_{ij}}{r_{ij}^3} \tag{339}$$

7.5 Artificial Viscosity

The SPH equations formulated above keep the specific thermodynamic entropy A_i strictly constant.

The Euler equations, however, can produce true discontinuities in form of shock waves or contact discontinuities[13] where the specific entropy increases at the shock front - which our current SPH scheme will never display. We must introduce artificial viscosity for the necessary dissipation processes producing heat and entropy to be possible. The artificial viscosity dampening the motion of particles broadens the shock layer into a differentially resolvable form. Also, without artificial viscosity, particles would be able to interpenetrate. We want the viscosity to only be active at shocks and not disturb our ideal gas behavior.

7.5.1 Viscous Pressure

Based on the discretized estimate $\langle \underline{\nabla} \cdot \underline{v} \rangle_i$, viscosity can be added in form of the following pressure (von-Neumann-Richtmyer-Landshoff)

$$p_i^{AV} = \begin{cases} \underbrace{-\alpha_i^{AV} \rho_i c_i h_i (\underline{\nabla} \cdot \underline{v})_i}_{\substack{\text{combined shear and} \\ \text{bulk viscosity,} \\ \text{dampens post shock} \\ \text{osci.}}} + \underbrace{\beta_i^{AV} \rho_i h_i^2 (\underline{\nabla} \cdot \underline{v})_i^2}_{\substack{\text{Richtmyer} \\ \text{viscosity, prevent} \\ \text{interpenetration} \\ \text{in high Mach} \\ \text{number shocks}}} & \text{if } (\underline{\nabla} \cdot \underline{v})_i < 0, \\ 0 & \text{otherwise} \end{cases} \tag{340}$$

density ρ, sound speed c, smoothing length h, parameters α, β

7.5.2 Adding the artificial viscosity to the equation of motion

To our SPH Euler equation, we can add the viscous force as

$$\underline{a}_i^{\text{visc}} = \left. \frac{d_{v_1}}{dt} \right|_{visc} = -\sum_{j=1}^{N_i} m_j \Pi_{ij} \cdot \underline{\nabla}_i \overline{W_{ij}}$$
$$\overline{W_{ij}} = \frac{1}{2} W_{ij} \left(\frac{h_i + h_j}{2} \right), \quad \Pi_{ij} \text{ should be symmetric } \rightarrow \text{ antisymmetric force} \tag{341}$$

where keeping the force antisymmetric retains conservation of linear and angular momentum.

[13] At the shock front the differential form of the Euler equation breaks down and the integral form leading to the Rankine-Hugoniot jump conditions has to be used.

For the viscous stress tensor we can model

$$
\text{one possibility } \Pi_{ij} = \begin{cases} \left[-\alpha c_{ij}\mu_{ij} + \beta\mu_{ij}^2\right]/\rho_{ij} & \text{if } \underline{v}_{ij}\cdot\underline{r}_{ij} < 0 \\ 0 & \text{otherwise} \end{cases}, \quad \mu_{ij} = \frac{h_{ij}\underline{v}_{ij}\cdot\underline{r}_{ij}}{\left|\underline{r}_{ij}\right|^2 + \epsilon h_{ij}^2}
$$
$$
\text{mean sound speed } c_{ij} = \frac{c_i + c_j}{2}, \quad \text{singularity protection } \epsilon \simeq 0.01
$$
$$
\text{viscosity strength regulated by } \alpha \simeq 0.5 to 1, \beta \simeq 2\alpha \tag{342}
$$

This form of a viscous force is a combination of a bulk and von-Neumann-Richtmyer viscosity and only acts if two particles (rapidly) approach [14] each other. It is Galilean invariant and vanishes for rigid body rotation.

The total momentum equation then is

$$
\frac{d\underline{v}_i}{dt} = -\sum_{j=1}^{N_i} m_j\left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} + \Pi_{ij}\right)\underline{\nabla}_i W\left(r_{ij}, h_{ij}\right) - \underline{\nabla}\phi_i \tag{343}
$$

In order to conserve total energy, work done against the viscous force has to show up as heat (discussed later).

7.5.3 Shear-Flow-Balsara correction

Problem: In the above we will also add high viscosity to shear flows (which we do not want), where particles also quickly approach each other (move adjacently with different velocities)

Idea: Such shear flows are marked by high vorticity $\underline{\omega} = \underline{\nabla} \times \underline{v}$, so for high vorticity we can crank down the artificial viscosity.

$$
\tilde{\mu}_{ij} = \mu_{ij}\cdot\frac{f_i + f_j}{2}, \quad f_i = \tag{344}
$$

7.5.4 Further viscosity switches

To reduce viscosity far away from shocks many other switches have been proposed, e.g. by making the viscosity strength parameters α variable in time and adopting $\beta = 2\alpha$.

[14] $\underline{v}_{ij}\cdot\underline{r}_{ij} < 0$, and μ_{ij} measure how strongly two particles approach each other.

Idea: We only want high $\Pi_i j$ when there is high compression, $\underline{\nabla} \cdot \underline{v}$ strongly negative, which we can use as a switch after which we let α_i decay exponentially.

7.6 SPH energy equation with artificial viscosity

The work done against the viscous force can be accounted in terms of energy or entropy.

Let us start with the hydrodynamic energy equation

$$\frac{d\epsilon}{dt} = \frac{\partial \epsilon}{\partial t} + \underline{v} \cdot \underline{\nabla} \epsilon = \frac{ds}{dt} - \frac{p}{\rho} \underline{\nabla} \cdot \underline{v} \tag{345}$$

from which for adiabatic systems with $\frac{ds}{dt} = 0$ and with eq. 334 we find

$$\frac{d\epsilon_i}{dt} = \frac{p_i}{\rho_i^2} \sum_{j=1}^{N_i} m_j \underline{v}_{ij} \cdot \underline{\nabla}_i W\left(r_{ij}, h_{ij}\right) \tag{346}$$

(where here we do not take the symmetric form as it can lead to unphysical solutions like negative internal energy).

To this we add a dissipation term due to artificial viscosity and incorporate heating and cooling sources into a function Γ_i to obtain

$$\boxed{\frac{d\epsilon_i}{dt} = \underbrace{\frac{p_i}{\rho_i^2} \sum_{j=1}^{N_i} m_j \underline{v}_{ij} \cdot \underline{\nabla}_i W_{ij}}_{\text{from hydrodynamic energy equation}} + \underbrace{\frac{1}{2} \sum_{j=1}^{N_i} m_j \Pi_{ij} \underline{v}_{ij} \cdot \underline{\nabla}_i W_{ij}}_{\text{dissipation from art. viscosity}} + \underbrace{\Gamma_i}_{\text{heating and cooling}}} \tag{347}$$

7.7 SPH Entropy equation

Alternatively to the energy equation, one can integrate an equation for the specific thermodynamic entropy A_i. The entropic function is defined by

$$P = A(s)\rho^\gamma, \quad \text{adiabatic index } \gamma \tag{348}$$

from which the internal energy follows as

$$\epsilon = \frac{P}{(\gamma - 1)\rho} = \frac{A(s)}{\gamma - 1} \rho^{\gamma - 1} \tag{349}$$

The SPH entropy equation can be derived as

$$\frac{\mathrm{d}A_i}{\mathrm{d}t} = -\frac{\gamma-1}{\rho_i}\Gamma_i + \frac{1}{2}\frac{\gamma-1}{\rho_i^{\gamma-1}}\sum_{\mathrm{J}=1}^{N_i} m_j \Pi_{ij} \underline{v}_{ij} \cdot \underline{\nabla}_i W\left(r_{ij}, h_{ij}\right) \tag{350}$$

From this using eq. 349 we can retrieve the internal energy and the temperature T_i (proportional to ϵ_i).

7.8 Maximum timestep - CFL criterion

A possible criterion is

$$\Delta t_i^{CFL} = 0.3\frac{h_i}{h\left|(\underline{\nabla}\cdot\underline{v})_i\right| + c_i + 1.2\left(\alpha_i c_i + \beta_i h_i \left|\min\left((\underline{\nabla}\cdot\underline{v})_i, 0\right)\right|\right)} \tag{351}$$
$$\text{sound speed } c_i, \quad \text{viscosity strength } \alpha_i, \beta_i$$

7.9 Notes on boundary modeling*

The two basic options are

- use fixed dummy particles (however those lead to the violation of the conservation of energy)
- use a fluid-solid force (e.g. inspired by the Lenard-Jones potential)

7.10 Reversibility in the context of viscosity-free, weakly-compressible SPH*

> Note that a fluid rising up to a dam form again after a dam-break and anti-dissipative or *clumping-up* so anti-pressure behavior are very different. The advantage of our SPH particles representing a fluid so acting on e.g. pressure unlike normal particles, the main advantage of rediscretizing based on the fluid equations, is of course sustained.

The SPH equations (based on the Euler fluid equations) lead us to a numerically time-reversible scheme[15] for the evolution of the positions and velocities of our SPH-particles - we must take care though and

- use a reversible, symplectic method like leapfrog
- calculate the density from the current positions of the particles not the evolution equation avoiding accumulation of density errors and making our SPH evolution symplectic.

[15] Solving backwards in time while recovering the initial conditions.

- use fixed-point over floating-point arithmetic, to avoid floating point errors violating reversibility

Note: The SPH-particles (without added viscosity) obey reversible Hamiltonian dynamics.

Indeed, for the N particle distribution function $f_N(t, \underline{r}_1, \underline{p}_1, \ldots, \underline{r}_N, \underline{p}_N)$ the Liouville entropy

$$S^{\text{Liouville}}\ (f_N) = -\frac{k_B}{N!} \int d\underline{r}_1 \int d\underline{p}_1 \cdots \int d\underline{r}_N \int d\underline{p}_N f_N \ln\left(h^{3N} f_N\right) \tag{352}$$

remains - in theory and in the reversible simulation (Kincl and Pavelka, 2023) - constant.

However, the Boltzmann entropy - which is obtained by maximizing the Liouville entropy under the constraint of the one-particle distribution $f(t, \underline{r}, \underline{p})$[16]

$$S^{\text{Boltzmann}}\ (f) = -k_B \int d\underline{r} d\underline{p} f \left(\ln\left(h^2 f\right) - 1\right) \tag{353}$$

grows in the simulation of the dam break experiment in Kincl and Pavelka, 2023 (with velocity distribution becoming Maxwellian).

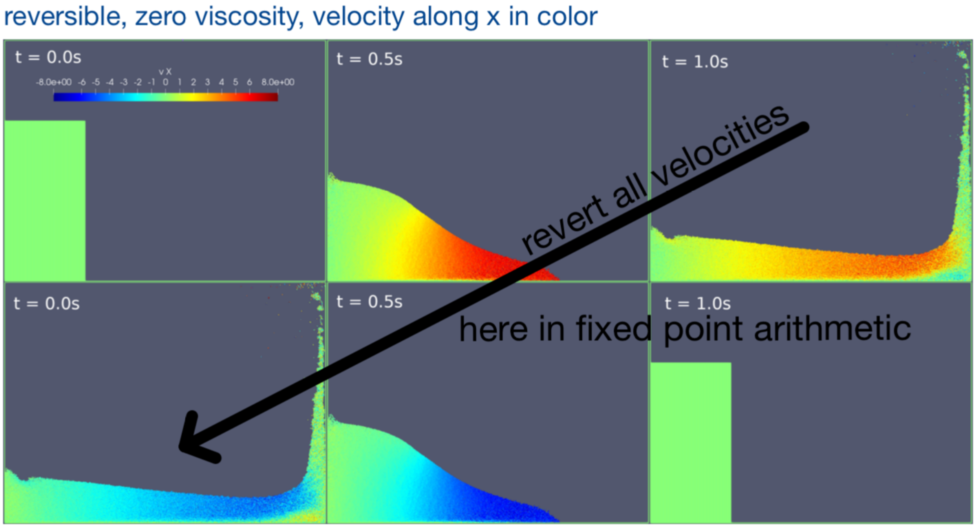
So forgetting the exact positions and momenta of our SPH particles, the one-particle density they describe increases in Boltzmann entropy, the 2nd law of thermodynamics emerges (statistically).

In the words from Kincl and Pavelka, 2023: »In summary, if we see all the positions and momenta in the SPH simulation, we can not see the second law of thermodynamics. Indeed, the simulation is reversible and the Liouville entropy remains constant. However, when we only focus on the one-particle distribution function, we can see the growth of Boltzmann entropy and thus irreversible behavior.«.

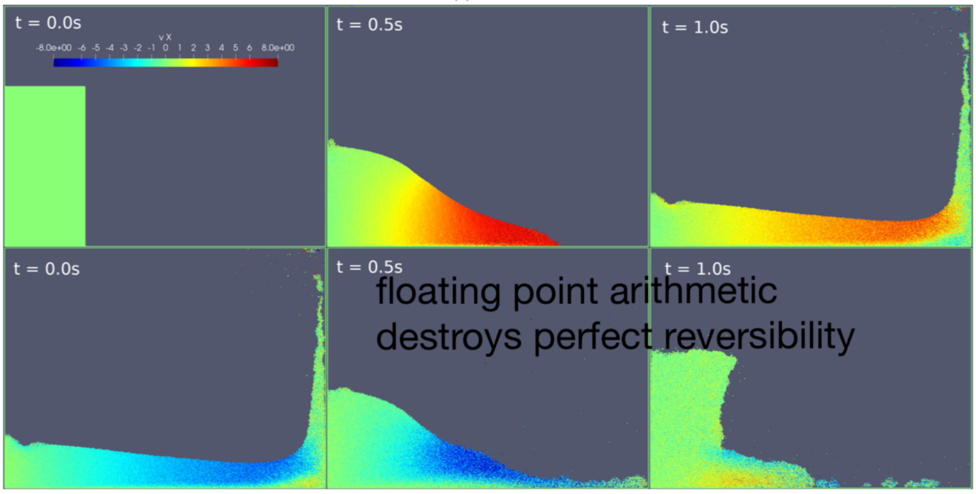
Note: Since we have a discrete (as of our discretization), deterministic, reversible system which can only existist in finitely many states, it is recurrent, thus will after long enough time come back to its initial state (unlikely though as of the high-dimensional phase space).

Note: The situation here is without any viscosity, so that if we invert the velocities and come back to our previous state (see figure 71) this is not anti-dissipation and does not happen when the system is dissipative (with viscosity). (?)

[16] Normalized to the number of particles.



(a) FixPA



(b) FloPA

Figure 71: Reversible SPH simulation without viscosity at the example of a breaking dam. In floating-point arithmetic this reversible behavior is broken. Being in an initial state leading back to the *dam* is very unlikely so even slight deviations will make us not go back to the dam.

Note the different levels at play: The SPH-particles are artificial in that they describe the fluid not the fluid particles. So while the SPH-particles start out resting, there is still temperature and pressure. However, in the dam experiment (fig. 71) (and generally) the SPH particles themselves thermalize to a Boltzmann distribution (they start out in non-equilibrium), fitting to the equilibrium Boltzmann entropy (although we do not add any heat, Boltzmann entropy of the SPH-particles (**based on their phase space distribution**) increases as we go from non-equilibrium to equilibrium). Note that this is not the same as the specific thermodynamic entropy of the fluid, as previously described.

7.11 Notes on the conservative formulation using Lagrange multipliers

Starting with the Lagrangian for inviscid, compressible flow

$$\mathcal{L} = \int \rho \left\{ \frac{1}{2} v^2 - u(\rho, s) \right\} d^3 r \tag{354}$$

the SPH equation with $s =$ const. follows from the Lagrange equation

$$\frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \underline{v}} - \frac{\partial \mathcal{L}}{\partial \underline{r}} = 0 \tag{355}$$

(for variable smoothing length h under the constraint (Lagrange multiplier) $\rho_j h_j^3 =$ const.) as

$$\frac{d\underline{v}_i}{dt} = -\sum_{j=1}^{N_i} m_j \left\{ \frac{1}{f_i} \frac{p_i}{\rho_i^2} \underline{\nabla}_i W\left(r_{ij}, h_i\right) + \frac{1}{f_j} \frac{p_j}{\rho_j^2} \underline{\nabla}_i W\left(r_{ij}, h_j\right) \right\}, \quad f_i = \left[1 + \frac{h_i}{3\rho_i} \frac{\partial \rho_i}{\partial h_i} \right] \tag{356}$$

From the symmetries of the Lagrangian it follows that energy, specific thermodynamic entropy, linear and angular momentum are conserved.

7.12 Further improvements

Ideas for improvement are

- it may be more physical to move particles with a smoothed flow velocity
- ...

7.13 Advantages and Disadvantages of SPH

Advantages and disadvantages of SPH are summarized in table 14.

Advantages	Disadvantages
• versatile and simple • automatically adaptive resolution, can cover large ranges in density and space • excellent conservation properties (energy, linear and angular momentum, not guarantueed in Eulerian codes) • inherent mass conservation • Galilean invariant and free from advection errors • can deal with complicated geometries as mesh-free • simple and transparent codes • very robust • as a particle scheme it is good for describing the transition from gaseous to stellar dynamical systems (e.g. formation of stellar clusters)	• limited accuracy in multi-D flow • low density regions are more poorly resolved • poorly handles shocks • noise as of discrete sums over limited neighbor set • jitter develops • fluid instabilities arcoss contact discontinuities are problematic such as Kelvin helmholtz instabilities • artificial viscosity limits the Reynolds number which can be reached • low convergence rate [17] • approximation of boundary conditions can be difficult • formation of voids • free surfaces can be problematic (density underestimated there) • magnetic fields are hard to handle (problems with stability and $\underline{\nabla} \cdot \underline{B} = 0$ requirement)

Table 14: Advantages and disadvantages of SPH.

8 Finite Element Methods

Finite element methods (FEM) are a class of methods for solving PDEs. Advantages include

- can work with flexible geometries
- handle geometrically intricate boundary conditions
- spatially adapt the resolution

Generally, numerical schemes need to represent a problem's solution by finitely many numbers, and then manipulate those numbers as true to the problem as possible. In finite volume methods we represent the solution as averages on cells which we update based on devised fluxes. In Smoothed Particle Hydrodynamics, we represent the fluid by a finite set of artificial *fluid particles* and update their positions and velocities according to the Lagrangian form of the fluid equations.

Idea of FEMs: In FEMs the solution domain is structured into finite elements with nodes on which base functions sit. The partial differential equation (or rather a weak form[a] of it) turns into equations for the (finitely many) weights of those basis functions.

[a]Weak here means, that the differential equation must not hold strictly locally, but for instance an integrated residual is to be zero, not the residual everywhere itself.

8.1 Finite element methods for linear PDEs

8.1.1 The solution is represented by weighted base functions on nodes within finite elements

The central ideas are

- Division of space: The space on which the solution sits is divided onto smaller regions called elements (e.g. segments in 1D, rectangles in 2D, cubes, octahedra, ... in 3D). Every element contains a certain number of points called nodes.
- Elementwise solution approximation: We element-wise approximate the solution with a set of linearly independent (not necessarily orthogonal) basis functions evaluated on nodes.

On an element we could for instance use a polynomial basis

$$\text{a line } \phi(x) = a_0 + a_1 x, \quad \text{or a parabula } \phi(x) = a_0 + a_1 x + a_2 x^2 \tag{357}$$

or Legendre polynomials.

Note: We need the same number of nodes as coefficients so that the values ϕ_i on the nodes fully specify our polynomial.

Better yet, we could use shape function N, so that our node values themselves are the coefficients we model.

So for n node values $\phi_1, \ldots, \phi_n$ (called *expansion coefficients*), we can write the solution on the k-th element as

$$\begin{gathered} \phi^{(k)}(x) = \phi_1^{(k)} N_1^{(k)}(x) + \phi_2^{(k)} N_2^{(k)}(x) + \cdots + \phi_n^{(k)} N_n^{(k)}(x) \\ \text{shape functions } N_i^{(k)}, \text{ zero outside k-th element} \end{gathered} \tag{358}$$

All elements use the same base function forms, but for each specific element the sum over its base functions is only non-zero in the respective element. We can therefore also write the total solution as

$$\phi(x) = \phi_1 N_1(x) + \phi_2 N_2(x) + \cdots + \phi_{\mathcal{N}} N_{\mathcal{N}}(x), \quad \text{in total } \mathcal{N} \text{ nodes} \tag{359}$$

Note: While we use 1D notation here, we can expand to $\phi(\underline{x}) = \sum_{i=1}^{n} \phi_i N_i(\underline{x})$. Time-dependency can be included as time-dependency of the weights aka expansion coefficients $\phi_i = \phi_i(t)$. But first we will consider constant coefficients, so a stationary (elliptic) problem.

Based on this the next step is turning the PDE into and algebraic equation for the expansion coefficients ϕ_i.

8.1.1.1 Example 1D linear reconstruction

A linear 1D example can be seen in figure 72.

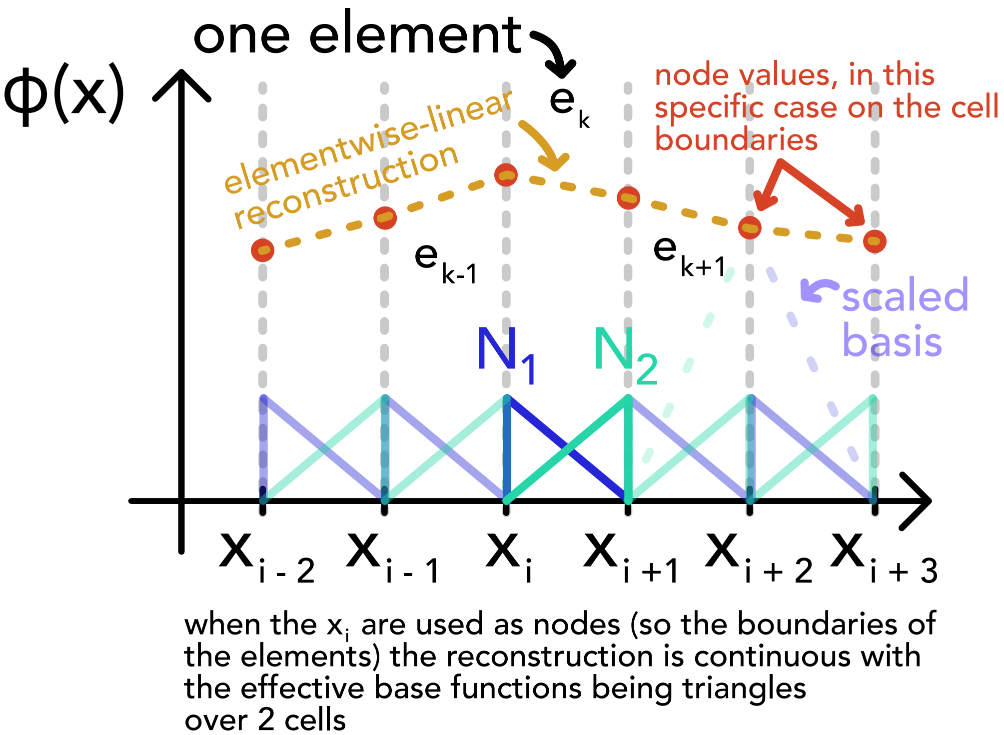


Figure 72: 1D linear reconstruction

8.1.2 From the PDE to an algebraic equation for the expansion coefficients $\phi_1, \ldots, \phi_n$

For now consider a stationary example (no time dependence of ϕ_j in one spatial dimension.)

In the following we find an algebraic expression (here even a system of linear equations) usually done for the expansion coefficients on one element. Afterwards the equations for all elements have to be assembled (to an overall linear system) in a further step. If the system has $\mathcal{N}$ total nodes, the final linear equation is described by an $\mathcal{N} \times \mathcal{N}$ system. See Lewis et al., 2004.

We might sometimes directly find such a system for all expansion coefficients and we will focus on that.

8.1.2.1 Inserting the finite element approximation into the PDE yields a residuum

Consider a linear PDE of the form

$$L\hat{\phi} + \hat{s} = 0, \quad \text{linear differential operator } \hat{L}, \text{ source term } \hat{s}, \text{ solution } \hat{\phi} \tag{360}$$

$\hat{L}$ being linear means $\hat{L}(a\phi + b\psi) = a\hat{L}\phi + b\hat{L}\psi$ for any functions ϕ, ψ and constants a, b.

We now plug in the finite element approximation $\phi(x)$ and source function $s(x)$

$$L\phi + s = R^{(k)}(x; \phi_1, \dots, \phi_n) = L\left(\sum_{j=1}^{n} \phi_j N_j(x)\right) + s \underbrace{=}_{L \text{ linear}} \sum_{j=1}^{n} \phi_j L N_j(x) + s \tag{361}$$

supscript k to indicate this is the residual over the k-th element

As only the shape functions depend on x, we only have to apply L to them. As of our approximation we have a generally non-zero residual.

8.1.2.2 Finding the expansion coefficients by minimizing the residual in some sense

We want to choose expansion coefficients minimizing the residual in some sense.

Ritz method: Here we require the integral over the residual to vanishes

$$\int_{\text{domain}} R\, dx = 0 \tag{362}$$

where here we consider the whole problem domain, to readily find the whole system of linear equations (but this might also be just an element). Therefore

$$R = R\left(x; \sum_{i=1}^{\mathcal{N}} \phi_i\right) \tag{363}$$

Weighted residual method Compared to the Ritz method weighting functions $w_i(x)$ are introduced

$$\int_{\text{domain}} w_i(x) R \, dx = 0, \quad i = 1, \dots, \mathcal{N} \tag{364}$$

which leads to

- collocation method: Residuum is required to vanish at n points x_i inside the domain, $w_i = \delta(x - x_i)$
- least-square method: $w_i = \partial_{\phi_i} R \to \int_{\text{domain}} (\partial_{\phi_i} R) R \, dx = \frac{1}{2} \partial_{\phi_i} \int_{\text{domain}} R^2 \, dx = 0$
- Galerkin method: Choose basis functions themselves as weights, so $w_i(x) = N_i(x)$, so

$$\int_{\text{domain}} N_i(x) R \, dx = 0 \tag{365}$$

The general Galerkin principle is to multiple an equation by arbitrary test functions and describe the unknown field with the same set of basis functions.

8.1.2.3 A linear system for $\phi_1, \dots, \phi_{\mathcal{N}}$ in the Galerkin scheme

Based on

$$\begin{aligned}
\forall i \in 1, \dots, \mathcal{N} : 0 = \int_{\text{domain}} N_i(x) R \left(\sum_j \phi_j N_j(x) \right) dx = \int_{\text{domain}} N_i(x) \cdot L \left(\sum_j \phi_j N_j(x) \right) + s dx \\
\underbrace{=}_{\text{L linear}} \sum_j \phi_j \underbrace{\int_{\text{domain}} N_i(x) L \left(N_j(x) \right) \, dx}_{A_{ij}} - \underbrace{\int_{\text{domain}} -N_i(x) s \, dx}_{b_i} \\
\to \sum_j \phi_j A_{ij} = b_i
\end{aligned} \tag{366}$$

we can turn the PDE into a linear system for the expansion coefficients ϕ_i, compactly

$$\underline{\underline{A}}\,\underline{\phi} = \underline{b}, \quad \underline{\phi} = \begin{pmatrix} \phi_1 \\ \vdots \\ \phi_{\mathcal{N}} \end{pmatrix}, \quad \text{vector of source elements } \underline{b} \tag{367}$$

$\underline{\underline{A}}$ very sparse because of the localization of the expansion elements

FEMs are only good for linear PDEs (only if L is linear do we get a linear system for the coefficients) (sometimes non-linear PDEs can be linearized though)

Dynamical systems where the ϕ_i might change in time will follow shortly.

8.1.2.4 Example Application of Galerkin FEM

Consider the 1D Poisson equation (really an ODE)

$$\partial_x^2 \phi(x) = 4\pi G \rho(x), \qquad \text{van Neumann boundary conditions } \left.\partial_x \phi\right|_{x_L} = \left.\partial_x \phi\right|_{x_R} = 0 \tag{368}$$

Aim: From a given density distribution ρ we want to find the field ϕ.

We formulate the basis functions as triangular basis functions spanning two elements, as shown in figure 72. In this figure you can also see that one can formulate this in terms of basis functions only sitting on one element, which is more true to the previous introduction.

$$S_i(x) = \begin{cases} \frac{x - x_{i-1}}{\Delta x} & \text{for } x \in [x_{i-1}, x_i] \\ \frac{x_{i+1} - x}{\Delta x} & \text{for } x \in [x_i, x_{i+1}] \\ 0 & \text{otherwise.} \end{cases} \tag{369}$$

We have linear elements between $\mathcal{N}$ points, Δx apart.

The finite element approximation is

$$\phi = \sum \phi_i S_i(x) \tag{370}$$

we calculate the residual as

$$R = \partial_x^2 \phi(x) - 4\pi G \rho(x) \tag{371}$$

so using the Galerkin weighting we get

$$\forall i = 1, ..., \mathcal{N} : \int_{x_L}^{x_R} S_i(x) \left(\partial_x^2 \phi(x) - 4\pi G \rho(x) \right) \, dx = 0 \rightarrow \int_{x_L}^{x_R} S_i(x) \partial_x^2 \phi(x) \, dx = \int_{x_L}^{x_R} S_i(x) 4\pi G \rho(x) \, dx \tag{372}$$

The LHS can be rewritten by integration over parts (mind $\left.\partial_x \phi\right|_{x_L} = \left.\partial_x \phi\right|_{x_R} = 0$)

$$\int_{x_L}^{x_R} S_i(x) \partial_x^2 \phi(x) dx = - \int_{x_L}^{x_R} \partial_x S_i(x) \partial_x \phi(x) dx = - \int_{x_L}^{x_R} S_i(x) 4\pi G \rho(x) dx = b_i \tag{373}$$

so the with the RHS

$$\int_{x_L}^{x_R} \partial_x S_i(x) \partial_x \sum \phi_j S_j(x) dx = \sum \phi_j \underbrace{\int_{x_L}^{x_R} \partial_x S_i(x) \partial_x S_j(x) dx}_{A_{ij}} = \sum \phi_j A_{ij} = b_i \tag{374}$$

we retrieve a linear equation with (use the definition of $S_i(x)$)

$$A_{ij} = \begin{cases} \frac{2}{\Delta x} \text{ for } i = j \\ -\frac{1}{\Delta x} \text{ for } i = j \pm 1 \\ 0 \text{ otherwise} \end{cases} \tag{375}$$

which would also follow from

$$\partial_x^2 \phi_i = \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2} \tag{376}$$

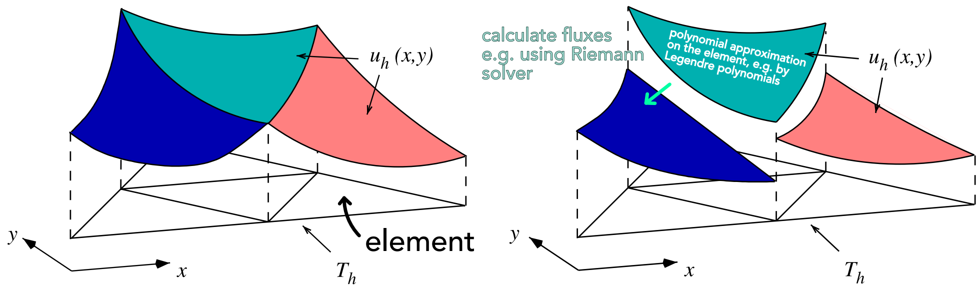
8.2 Discontinuous Galerkin Method

Aim: We want to find the solution to a hyperbolic differential equation system. The solution is often characterized by discontinuities and shocks. Me might be interested in complex geometries, for which Finite Element Methods are very suitable.

Problem: Usual Finite Element Methods (FEMs) are piecewise polynomial and continuous - shocks are often smeared out.

Idea: Combine the advantages of Finite Element Methods and Finite Volume Schemes. Discontinuous Galerkin is a FEM - the problem domain is subdivided into a grid of a finite number of elements. We use a piecewise polynomial solution which can be discontinuous across cell interfaces, where we use the methods from finite volume to compute the intercell fluxes - so conservation laws are baked in.

Continuous and discontinuous Galerkin are illustrated in figure 73.



(A) Continuous Galerkin (B) Discontinuous Galerkin

Figure 73: Continuous and discontinuous Galerkin

8.2.1 Problem we want to tackle | Euler equations

Consider the 3D formulation of the Euler equations we know from dimensional splitting

$$\partial_t \underline{u}(\underline{x}) + \sum_{\alpha=1}^{3} \partial_{x_\alpha} \underline{f}_\alpha(\underline{u}) = 0, \quad \text{state vector } \underline{u} = \begin{pmatrix} \rho \\ \rho \underline{v} \\ \rho e \end{pmatrix} \tag{377}$$

$$\text{specific energy } e = \rho e_{th} + \frac{1}{2}\rho \underline{v}^2 \text{ with specific internal energy } e_{th}$$

with flux vectors

$$\underline{f}_1 = \begin{pmatrix} \rho v_1 \\ \rho v_1^2 + P \\ \rho v_1 v_2 \\ \rho v_1 v_3 \\ v_1(\rho e + P) \end{pmatrix}, \quad \underline{f}_2 = \begin{pmatrix} \rho v_2 \\ p v_1 v_2 \\ \rho v_2^2 + P \\ \rho v_2 v_3 \\ v_2(\rho e + P) \end{pmatrix}, \quad \underline{f}_2 = \begin{pmatrix} \rho v_3 \\ p v_1 v_3 \\ \rho v_2 v_3 \\ \rho v_3^2 + P \\ v_3(\rho e + P) \end{pmatrix} \tag{378}$$

$$\text{ideal gas closure } P = \rho e_{th}(\gamma - 1)$$

Aim: From a given initial state $\underline{u}(\underline{x}, t = 0) = u(\underline{x}, 0)$ we want to find the subsequent evolution of the fluid.

8.2.2 Steps in formulating the Dicontinuous Galerkin (DG) scheme

1. Subdivide the space into elements aka cells
2. Represent the fluid state on a cell using a polynomial basis (e.g. Legendre polynomials) with weights evolving in time; *nodal* vs *modal*
3. Find a general formula for the weights in the *modal* variant
4. Find the initial weights from specific *nodal* starting values in the *modal* scheme

5. Find an evolution equation for the weights

8.2.3 Subdivision and Representation | modal vs nodal

The fluid state is represented as a polynomial approximation (for the respective fluid variables) on the element (non overlapping elements with discontinuities in-between) - but how?

A typical DG cell is illustrated in figure 74.

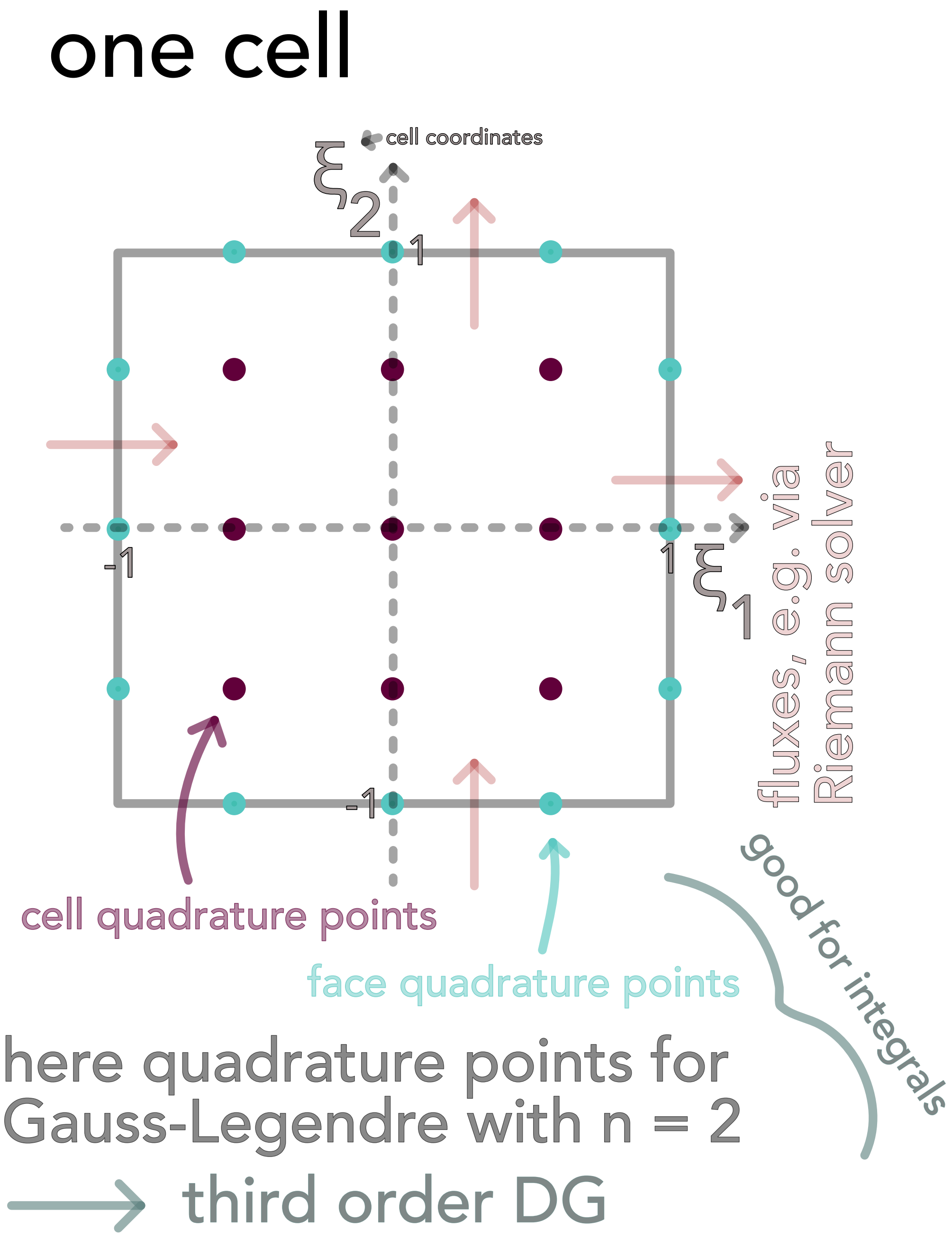


Figure 74: Typical DG cell

There are two possible representations of the solution

- **nodal**: We store and operate on fluid state vectors at chosen positions within the cell. In the Legrange interpolation with Lagrange polynomials of degree k, so $l_j(x) = \prod_{\substack{0 \le m \le k \\ m \ne j}} \frac{x - x_m}{x_j - x_m}$, for one fluid variable the node values are also the expansion coefficients, $u = \sum_{j=0}^{N(k)} u_j l_j(x)$. The positions of the nodes in the cell are chosen smartly regarding quadrature (integration) rules.
- **modal**: We store and operate on weights of usually orthogonal polynomials (usually Legendre). Integrals are still evaluated using quadrature rules. Initial weights have to be calculated based on nodal values.

We opt for **modal**.

The solution in the interior of cell K is given by a linear combination of $N(k)$ orthogonal and normalized basis functions $\phi_l^{(K)}$ (maximum degree of the basis functions is k).

$$\underline{u}^{(K)}(\underline{x}, t) = \sum_{l=1}^{N(k)} \underline{\omega}_l^{(K)}(t)\phi_l^{(K)}(\underline{x}) \tag{379}$$

where

- the space-time dependence of the fluid state was split into time dependent weights and space dependent basis functions
- the cell's state is completely given by the $N(k)$ weights

8.2.3.1 Example for an orthogonal polynomial basis: Legendre polynomials

We can combine Legendre polynomials to 3D base functions with degree up to k in the following manner

$$\begin{gathered} \{\phi_l(\underline{\xi})\}_{l=1}^{N(k)} = \left\{\tilde{P}_u(\xi_1)\,\tilde{P}_v(\xi_2)\,\tilde{P}_w(\xi_3) \mid u, v, w \in \mathbb{N}_0 \wedge u + v + w \le k\right\} \\ \text{scaled Legendre polynomials } \tilde{P}_n(\xi) = \sqrt{2n+1}P_n(\xi) \\ \text{"special orthog. "} \int_{-1}^{1} P_i(\xi)P_j(\xi)d\xi = \begin{cases} 0 & \text{if } i \ne j \\ 2 & \text{if } i = j \end{cases} \end{gathered} \tag{380}$$

For a polynomial basis with maximum degree k we have

$$N(k) = \sum_{u=0}^{k} \sum_{v=0}^{k-u} \sum_{w=0}^{k-u-v} 1 \tag{381}$$

basis polynomials.

The first Legendre polynomials are

$$P_0(\xi) = 1, P_1(\xi) = \xi, P_2(\xi) = \frac{1}{2}\left(3\xi^2 - 1\right) \tag{382}$$

Note: The span of $P_0(x), P_1(x), \ldots, P_n(x)$ is the same as of $1, x, x^2, \ldots, x^n$. So

$$\forall \text{polynomial } P \text{ of degree } < n : \int_{-1}^{1} P(x)P_n(x)\, dx = 0 \tag{383}$$

8.2.4 Solving for the weights

Note: The main application of finding weights is finding the weights of the initial state. Going from weights to evaluations is easy.

Let's say we would know $\underline{u}^{(K)}(\underline{x}, t)$ - then we can determine the weights based on the orthogonality and normalization properties of the basis functions as

$$\begin{gathered} \underline{\omega}_j^K(t) = \frac{1}{|K|}\int_K \underline{u}^{(K)}\phi_j^{(K)} dV, \quad j = 1, \ldots, N(k) \\ \text{with } |K| \text{ volume of cell } K \end{gathered} \tag{384}$$

We hereby choose $\phi_1^{(K)} = 1$ so that $\omega_1^{(K)}$ is the cell's average of the state vector $\underline{u}^{(K)}$ (constant term). $\phi_j^{(K)}, j \geq 2$ are higher order basis functions, with $w_j^{(K)}$ being the higher order moments of the state vector $\underline{u}^{(K)}$.

Scaled variable on cell: On a cube, we can define the basis functions in terms of a scaled variable $\underline{\xi}$.

$$\begin{gathered} \phi_l(\underline{\xi}) : [-1, 1]^3 \to \mathbb{R}, \quad \underline{\xi} = \frac{2}{\Delta x}\left(\underline{x} - \underline{x}^{(K)}\right) \\ \text{cell's center } \underline{x}^{(K)}, \text{ cell's sidelength } \Delta x \end{gathered} \tag{385}$$

Idea: Let us approximate this integral by a quadrature rule.

First write the integral equation for the weights in the reference frame of a cubic cell with sidelength 2

$$\underline{\omega}_j^{(K)}(t) = \frac{1}{8}\int_{[-1,1]^3} \underline{u}^{(K)}(\underline{\xi}, t)\phi_j^{(K)}(\underline{\xi}) d^3\underline{\xi}, \quad j = 1, \ldots, N(k) \tag{386}$$

We then apply Gauss-Legendre quadrature with $(k+1)^3$ quadrature points, so

$$\underline{\omega}_j^{(K)}(t) \approx \frac{1}{8} \sum_{q=1}^{(k+1)^3} \underline{u}^{(K)} \left(\underline{\xi}_q^{3D}, t\right) \phi_j^{(K)} \left(\underline{\xi}_q^{3D}\right) w_q^{3D}, \quad j = 1, \ldots, N(k)$$
$$\text{positions of quadrature nodes in cell's reference frame } \underline{\xi}_q^{3D}, \text{ quadrature weights } w_q^{3D} \tag{387}$$

8.2.4.1 What even is Gauss-Legendre quadrature?*

It is intuitive that an integral can be approximated by the mean of evaluation points. However, it turns out, that one can exactly integrate polynomials of degree $2n-1$ with n smart evaluation points and weights. One such method is called Gauss-Legendre quadrature.

Claim: The approximation

$$\int_{-1}^{1} f(\xi) d\xi \approx \sum_{q=1}^{n} f\left(\xi_q^{1D}\right) w_q^{1D}$$
$$\text{roots } \xi_q^{1D} \text{ of } P_n(\xi), \text{ weights } w_q^{1D} = \frac{2}{\left(1 - \left(\xi_q^{1D}\right)^2\right) \left(P_n'\left(\xi_q^{1D}\right)\right)^2} \tag{388}$$

for $f : [-1, 1] \to \mathbb{R}$ is exact for polynomials of degree $2n-1$.

Proof: Let $f = P(x)$ have degree $\leq 2n-1$. Then we can write $P(x) = Q(x)P_{n+1}x + R(x)$ with $Q(x), R(x)$ polynomials of degree $\leq n$ ($P_{n+1}(x)$ is the $(n+1)$-th Legendre polynomial). Then

$$\begin{aligned}
\int_{-1}^{1} P(x)dx &= \underbrace{\int_{-1}^{1} Q(x)P_{n+1}(x)dx}_{=0 \text{ as } Q \text{ can be written in base } \{P_0,\ldots,P_n\} \perp P_{n+1}} + \int_{-1}^{1} R(x)dx \\
&= \int_{-1}^{1} R dx \\
&= \underbrace{\sum_{q=1}^{n} R\left(\xi_q^{1D}\right) w_q^{1D}}_{\text{exact as } R \text{ is a polynomial of degree } \leq n} \\
&= \sum_{q=1}^{n} \left(R\left(\xi_q^{1D}\right) + \underbrace{P_{n+1}(\xi_q^{1D}) \cdot Q(\xi_q^{1D})}_{=0, \text{as } \xi_q^{1D} \text{ roots of } P_{n+1}} \right) w_q^{1D} \qquad = \sum_{q=1}^{n} P(\xi_q^{1D}) w_q^{1D}
\end{aligned} \tag{389}$$

Formulation in higher dimensions: We generalize to $f : [-1,1]^2 \to \mathbb{R}$ by

$$\int_{-1}^{1}\int_{-1}^{1} f\left(\xi_1,\xi_2\right) d\xi_1 d\xi_2 \approx \sum_{q=1}^{n}\sum_{r=1}^{n} f\left(\xi_q^{1D},\xi_r^{1D}\right) w_q^{1D} w_r^{1D} = \sum_{q=1}^{n^2} f\left(\underline{\xi}_q^{2D}\right) w_q^{2D} \tag{390}$$

8.2.5 Finding initial weights - just apply the determination of weights to the initial state

The initial state $\underline{u}(\underline{x}, t=0)$ is best represented by weights $\underline{\omega}_j^{(K)}(t=0)$, such that

$$w_{l,i}^{(K)}(t=0) = \underset{\underline{\omega}_{j,i}^{(K)}(t=0)}{\operatorname{argmin}} \int_K \left(u_i^{(K)}(\underline{x}, t=0) - u_i(\underline{x}, t=0)\right)^2 d^3\underline{x}, \quad i \text{ over state vector components} \tag{391}$$

which just leads us to the previous projection (eq. 384) and solution in eq. 387 at $t = 0$, where our initial state must be known on the quadrature nodes.

8.2.6 Evolution equation for the weights

We derive a DG scheme on cell K.

Weak form of the Euler equations: A weak formulation of the Euler equations for the polynomial approximation $\underline{u}^{(K)}$ on cell K is found by multiplying the Euler equations with the basis function $\phi_j^{(K)}$ and integrating over the cell K.

$$\int_K \left[\partial_t \underline{u}^{(K)} + \sum_{\alpha=1}^{3} \partial_{x_\alpha} f_\alpha\right] \phi_j^{(K)} dV = 0 \tag{392}$$

Integrating by parts and applying Gauss theorem, we get

$$\frac{d}{dt}\underbrace{\int_K \underline{u}^{(K)}\phi_j^K dV}_{\underline{w}_j^{(K)}\cdot|K|} + \sum_{\alpha=1}^{3}\underbrace{\int_{\partial K} f_\alpha n_\alpha \phi_j^K dS}_{\substack{\text{evaluate using}\\ \text{Gauss-Quad,}\\ \text{unknown flux}\\ \text{across discont.}\\ \text{via Riemann}\\ \text{solver}}} - \sum_{\alpha=1}^{3}\underbrace{\int_K \underline{f}_\alpha \frac{\partial \phi_j^K}{\partial x_\alpha} dV}_{\substack{\text{evaluate via}\\ \text{Gauss-Quad.,}\\ \text{interior flux}\\ \text{known from}\\ \text{state variable}\\ \text{approx.}}} = 0 \tag{393}$$

$$\text{normal vector } \underline{n} = \begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} \text{ on } \partial K$$

- we have found a system of coupled ODEs for the weights which can be solved for instance using RK schemes. Remember to transform to the cell coordinates $\left(d\xi^3 = \left(\frac{2}{\Delta x^{(K)}}\right)^2 dV\right)$ (from eq. 385).

Note on strong shocks: In case there are strong shock waves, limiting-schemes damping oscillations (e.g. by setting higher expansion coefficients in cells adjacent to the detected discontinuity to zero) are used.

8.2.7 Efficiency of DG and refinement schemes

Accuracy can be increased by

- p-refinement: higher order scheme (polynomials up to higher order degree in the basis set)
- h-refinement: finer grid with smaller spacing

In both cases, the number of weights increases. For isentropic vortex flow, one finds that p-refinement is much more efficient (better solution at less CPU time) than h-refinement.

more on nodal vs modal, more details from script

9 Diffusion

On a micro scale diffusion is a random walk with the mean free path as the step size. Intuitive example: Imagine a region with high density. It is more likely that a particle within this region takes a step out of it than a particle outside stepping in (as of the difference in density). Physically, the entropy cannot decrease.

9.1 Thermodynamic Basics of Diffusion

> How quickly do thermalized, randomly moving particles spread?

9.1.1 Mean squared velocity and mean squared relative velocity

Assume the particles follow a Maxwell-Boltzmann distribution (we assume only one particle species with mass m and temperature T)

$$\begin{aligned} f(\underline{v})d^3v &= \left(\frac{m}{2\pi k_B T}\right)^{\frac{3}{2}} \exp\left(\frac{-\frac{1}{2}mv^2}{k_B T}\right) d^3v \\ f(v)dv &= 4\pi v^2 \left(\frac{m}{2\pi k_B T}\right)^{\frac{3}{2}} \exp\left(\frac{-\frac{1}{2}mv^2}{k_B T}\right) dv, \quad v = |\underline{v}| \end{aligned} \tag{394}$$

Based on the equipartition theorem, the mean squared velocity is

$$\langle \frac{1}{2} m\underline{v}^2 \rangle = \frac{3}{2} k_B T \quad \rightarrow \quad \langle \underline{v}^2 \rangle = \frac{3k_B T}{m}, \quad v := \sqrt{\langle \underline{v}^2 \rangle} \tag{395}$$

so the mean squared relative velocity is higher (avg here over velocity space)

$$\langle \underline{v}_{rel}^2 \rangle = \langle (\underline{v} - \underline{v}')^2 \rangle = \langle \underline{v}^2 \rangle + \langle \underline{v}'^2 \rangle - \underbrace{2\langle \underline{v} \cdot \underline{v}' \rangle}_{=0 \text{ avg. over uncorrelated}} = \langle \underline{v}^2 \rangle = \frac{6k_B T}{m} \tag{396}$$

where we shorthand use $v_{rel} = \sqrt{\langle \underline{v}_{rel}^2 \rangle}$.

9.1.2 Mean free path and relaxation time

Consider a fluid with particle density n (in m^{-3}) of particles with effective radius r so effective cross-section $\sigma = \pi r^2$. In sec. 5.1.2.1, we already devised (under particle movement with the root-mean-square velocity)

$$\begin{aligned} \text{particle moving through stationary particles: } \lambda_{mfp} &= \frac{1}{n\sigma} \\ \text{relative particle movement: } \lambda_{mfp} &= \frac{1}{\sqrt{2}n\sigma} \end{aligned} \tag{397}$$

and the relaxation time (mean time until a particle collides with another)

$$t_{rel} = \frac{\lambda_{mfp}}{v} = \frac{1}{n\pi r^2}\sqrt{\frac{m}{6k_BT}} \tag{398}$$

Springel et al., 2023 - I would say wrongly - uses $\lambda_{mfp} = \frac{1}{n\sigma}$ and with this defines $t_{scat} = \frac{\lambda_{mfp}}{v}$ and $t_{rel} = \frac{\lambda_{mfp}}{v_{rel}}$, which is equivalent to our t_{rel} with our λ_{mfp}.

9.1.3 Random Walk | spreading Gaussian distribution in space

Central Limit Theorem: Let $w(s)ds$ be the probability of taking a step with length between s and $s+ds$ with mean $\langle s\rangle$ and variance σ_s^2. We make N steps, so the final position along one dimension is

$$x = s_1 + s_2 + s_3 + \cdots + s_N \tag{399}$$

By the rules of the mean and variance, one finds

$$\begin{aligned} \langle x\rangle = N\langle s\rangle, \quad \langle s\rangle = \int_{-\infty}^{\infty} s\cdot w(s)\,ds \\ (\Delta x)^2 = \sigma_x^2 = \langle (x-\langle x\rangle)^2\rangle = N\sigma_s^2 \end{aligned} \tag{400}$$

And the **Central Limit Theorem** states

$$\boxed{P(x) = \frac{1}{\sqrt{2\pi}\Delta x}\exp\left(-\frac{(x-\langle x\rangle)^2}{2(\Delta x)^2}\right)} \tag{401}$$

if the moments $\langle s^n\rangle = \int_{-\infty}^{\infty} s^n w(s)\,ds$ are finite. The basis reasoning is that one can show that for added random variables their sum converges to one distribution, and as added random variables are distributed according to the convolution of their distributions and the convolution of two Gaussians is a Gaussian, the sum of many random variables is a Gaussian.

Consider particles starting out at the origin with equal probability of going left or right with λ_{mfp}. We have

$$w(s) = \frac{1}{2}\delta(s-\lambda_{mfp}) + \frac{1}{2}\delta(s+\lambda_{mfp}), \quad \langle s\rangle = 0, \quad \sigma_s^2 = \lambda_{mfp}^2 \tag{402}$$

so therefore, estimating $N = \frac{t}{t_{rel}}$

$$\langle x \rangle = 0, \quad \langle x^2 \rangle = N\sigma_s^2 = \frac{\lambda_{mfp}^2}{t_{rel}} t \to \boxed{\sqrt{\langle x^2 \rangle} \propto \sqrt{t}} \tag{403}$$

So the distribution flattens and spreads in time. As $\langle (\Delta x)^2 \rangle$ is time dependent, so is $p(x,t)$.

9.2 Diffusion equation

9.2.1 Derivation of the diffusion equation | Fick's law from the microscopic consideration

Consider the evolution of a system from $t \to t+\Delta t$. For a change Δt let Δx be the according change of a particle position, distributed according to $p(\Delta x, \Delta t)$. We can therefore write

$$\begin{aligned} n(x, t+\Delta t) = \langle n(x-\Delta x, t) \rangle &\underset{\text{Taylor}}{=} n(x,t) - \partial_x n \cdot \langle \Delta x \rangle + \frac{1}{2}\partial_x^2 n \cdot \langle \Delta x^2 \rangle + \dots \\ &\underset{\text{vanishing uneven moments}}{=} n(x,t) + \frac{1}{2}\partial_x^2 n \cdot \langle \Delta x^2 \rangle + \mathcal{O}(\Delta x^4) \end{aligned} \tag{404}$$

Taylor expansion with respect to time yields

$$n(x, t+\Delta t) = n(x,t) + \partial_t n \cdot \Delta t \tag{405}$$

So combined, we get (ignoring the higher order terms, so it's really an approximation)

$$\boxed{\partial_t n(x,t) = \frac{\Delta x^2}{2\Delta t}\partial_x^2 n(x,t) = D\partial_X^2 n(x,t)} \tag{406}$$

with

$$\text{diffusion coefficient } D = \frac{\Delta x^2}{2\Delta t} \propto \frac{\lambda_{mfp}^2}{t_{rel}} \tag{407}$$

We can rewrite this as a conservation equation (introducing the flux J) (Fick's law of diffusion)

$$\partial_t n(x,t) = -\partial_x J(x,t), \quad J(x,t) = -D\partial_x n(x,t) \tag{408}$$

where the same principle applies to heat, energy and momentum diffusion.

9.2.2 Analytical Solution to the diffusion equation via Fourier transform

Using the Fourier transform in space in the convention

$$n(x,t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \hat{n}(k,t)\exp(-ikx)dk, \quad \hat{n}(k,t) = \int_{-\infty}^{\infty} n(x,t)\exp(ikx)dx \tag{409}$$

we yield an ODE in Fourier space with trivial solution

$$\partial_t \hat{n}(k,t) = -Dk^2\hat{n}(k,t) \quad \to \quad \hat{n}(k,t) = \hat{n}\left(k,t_0\right)\exp\left(-Dk^2\left(t-t_0\right)\right) \tag{410}$$

9.2.2.1 Solution for an initial delta peak in the density over space

Consider

$$n\left(x,t_0\right) = n_0\delta\left(x-x_0\right), \quad \hat{n}\left(x,t_0\right) = n_0\int_{-\infty}^{\infty}\delta\left(x-x_0\right)\exp(ikx)dx = n_0\exp\left(ikx_0\right) \tag{411}$$

yielding the solution in Fourier space

$$\to \hat{n}(k,t) = \hat{n}\left(k,t_0\right)\exp\left(-Dk^2\left(t-t_0\right)\right) = n_0\exp\left(ikx_0\right)\exp\left(-Dk^2\left(t-t_0\right)\right) \tag{412}$$

so with the inverse Fourier transform (completing the square and using that a Gaussian distribution is normed)

$$\text{a Gaussian } n(x,t) = \frac{n_0}{\sqrt{4\pi D\left(t-t_0\right)}}\exp\left(-\frac{\left(x-x_0\right)^2}{4D\left(t-t_0\right)}\right) \tag{413}$$

Conservation: As the Gaussian is normalized, n_0 stays constant irrespective of time implying particle number conservation (or energy conservation for the diffusion of heat (heat conduction))

Problem of particle propagation: As the solution is a Gaussian with infinite wings, there must have been infinitely fast particle transport starting from the initial situation - unphysical. **Reason - approximation in diffusion equation**: Our diffusion equation is based on a random walk with well-behaved steps but in finding the diffusion equation we have done a truncated Taylor expansion, leaving away $\mathcal{O}(\Delta x^4)$. In numerical schemes we will use limited flux.

Infinite Signal Speeds: In contrast to advection with a given signal speed the diffusion equation and also Poisson equation (e.g. for Gravity) have infinite domains of influence - an event at $(\underline{x}, t)$ depends on the fill domain. In the case of diffusion we will later introduce limited flux aka tempered diffusion.

9.3 Numerical solutions

Let us change the notation $n \to u$ to be consistent with previously introduced schemes. We indicate with the convention $u_{\text{space}}^{(\text{time})}$.

In the following we discuss multiple possible discretizations.

The general grid in space and time is illustrated in figure 75.

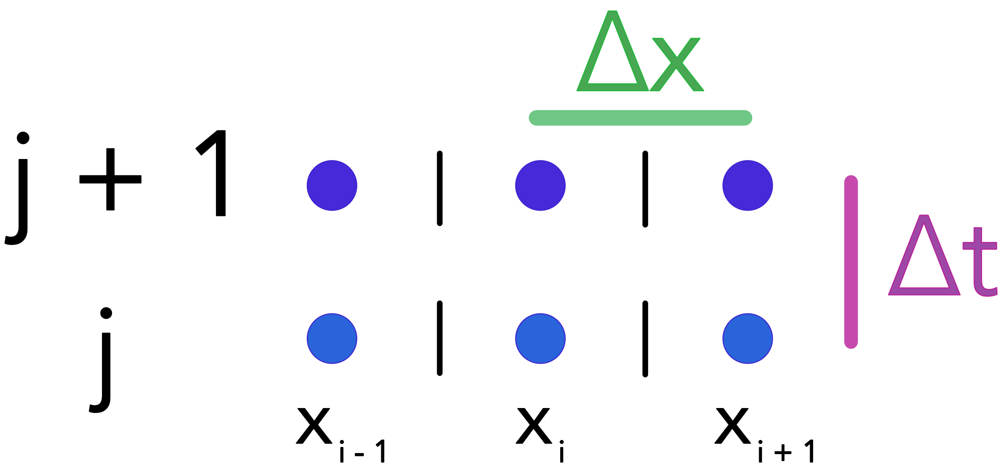


Figure 75: Grid in space and time for the diffusion equation.

9.3.1 Forward in time, central in space

9.3.1.1 Discretized diffusion equation

Based on the central approximation of $\partial_x^2 u$ (from Taylor in both directions, see eq. 244) at the current time j and forward difference in time, we get

$$\frac{u_i^{(j+1)} - u_i^{(j)}}{\Delta t} = D \frac{u_{i+1}^{(j)} - 2u_i^{(j)} + u_{i-1}^{(j)}}{\Delta x^2} \tag{414}$$

9.3.1.2 Explicit scheme for performing a time step

Defining $\alpha = \frac{D\Delta t}{\Delta x^2}$, we get

$$u_i^{(j+1)} = u_i^{(j)} + \alpha \left(u_{i+1}^{(j)} - 2u_i^{(j)} + u_{i-1}^{(j)} \right) \tag{415}$$

an explicit scheme for forwarding. The information flow is illustrated in figure 76.

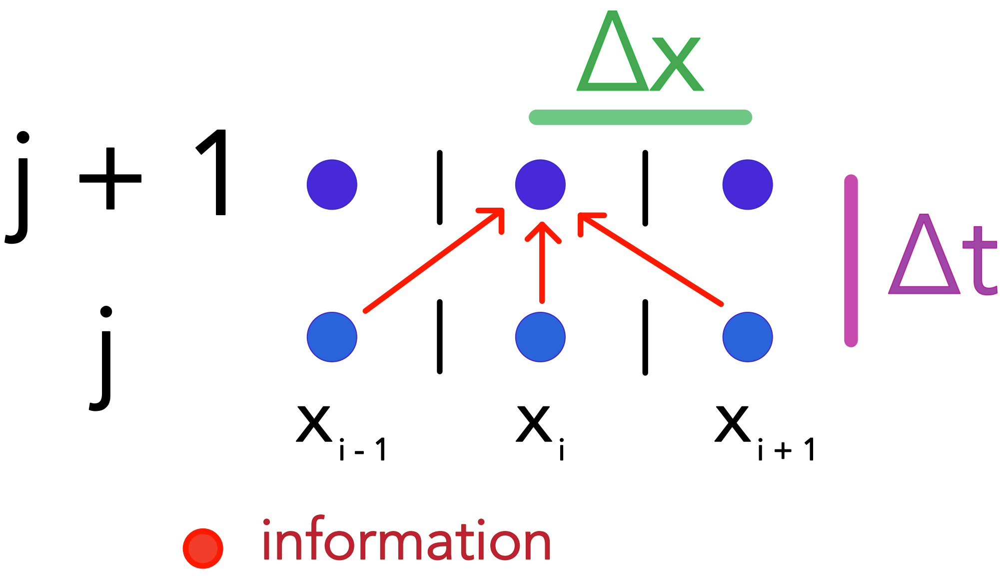


Figure 76: Information flow for the central in space, forward in time scheme.

9.3.1.3 Stability of the central in space, forward in time scheme for diffusion

Let us make a simple stability analysis considering $u_{i+1}^{(j)} = u_{i-1}^{(j)} = 0$. In such a scenario it does not make sense for there to be a sign change from $u_i^{(j)}$ to $u_i^{(j+1)}$. This requires (just plug conditions in)

$$\alpha \leq \frac{1}{2} \leftrightarrow \Delta t \leq \frac{1}{2}\frac{\Delta x^2}{D} \tag{416}$$

Physical intuition and connection to CFL: In diffusion, density spreads with $\Delta x = \sqrt{2D\Delta t}$, so when we impose a CFL criterion - we only use information from the left and right grid points next to the grid point of interest, so Δt must not be so large, that further information would in reality be necessary - we get the same criterion as above.

Problem of scaling the resolution: If we want to double the spatial resolution, we must quadruple the resolution in time, i. e. in total the 1D simulation is 8x more expensive (in 3D double spatial resolution means 8x more points in a certain volume so 32x more cost in total).

9.3.2 Backward in time, central in space

9.3.2.1 Discretized implicit scheme

We now evaluate the spatial derivative at time $j+1$ instead of j.

$$\frac{u_i^{(j+1)} - u_i^{(j)}}{\Delta t} = D\frac{u_{i+1}^{(j+1)} - 2u_i^{(j+1)} + u_{i-1}^{(j+1)}}{\Delta x^2} \tag{417}$$

This can be rewritten as

$$-\alpha u_{i+1}^{(j+1)} + (1+2\alpha)u_i^{(j+1)} - \alpha u_{i-1}^{(j+1)} = u_i^{(j)} \tag{418}$$

- a linear system of equations for $u_i^{(j+1)}$.

9.3.2.2 Matrix equation

The exact matrix depends on the boundary conditions we choose. For periodic boundary conditions[18] we get

$$\underline{\underline{A}}\underline{u}^{(j+1)} = \underline{u}^{(j)}, \quad \underline{u}_i^{(j)} = \begin{pmatrix} u_1^{(j)} \\ u_2^{(j)} \\ \vdots \\ u_N^{(j)} \end{pmatrix}$$

$$\underline{\underline{A}} = \begin{pmatrix} 1+2\alpha & -\alpha & 0 & \dots & & 0 & -\alpha \\ -\alpha & 1+2\alpha & -\alpha & 0 & \dots & & \\ 0 & -\alpha & 1+2\alpha & -\alpha & 0 & \dots & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \ddots & \ddots & \ddots \\ -\alpha & 0 & \dots & & 0 & -\alpha & 1+2\alpha \end{pmatrix} \tag{419}$$

which we can solve for $\underline{u}^{(j+1)} = \underline{\underline{A}}^{-1}\underline{u}^{(j)}$.

Problem: This is unconditionally stable but only first order in time $\mathcal{O}(\Delta x^2, \Delta t)$.

9.3.3 Crank-Nicolson method

Idea: Evaluate the spatial derivative at $j + \frac{1}{2}$, constructed as an average of j and $j+1$.

[18] The point before the first point is the last point and the point after the last point is the first point.

$$\frac{u_i^{(j+1)} - u_i^{(j)}}{\Delta t} = D\frac{u_{i+1}^{(j+\frac{1}{2})} - 2u_i^{(j+\frac{1}{2})} + u_{i-1}^{(j+\frac{1}{2})}}{\Delta x^2}, \quad \text{use avg. } u_i^{(j+\frac{1}{2})} = \frac{u_i^{(j+1)} + u_i^{(j)}}{2} \tag{420}$$

yielding

$$-\alpha u_{i+1}^{(j+1)} + (1+2\alpha)u_i^{(j+1)} - \alpha u_{i-1}^{(j+1)} = \alpha u_{i+1}^{(j)} + (1-2\alpha)u_i^{(j)} + \alpha u_{i-1}^{(j)} \tag{421}$$

So again a matrix problem over the whole spatial domain

$$\underline{\underline{A}}\,\underline{u}^{(j+1)} = \underline{b}, \quad \underline{\underline{A}} \text{ as before}, \quad b_i = \alpha u_{i+1}^{(j)} + (1-2\alpha)u_i^{(j)} + \alpha u_{i-1}^{(j)} \tag{422}$$

where again we can solve for $\underline{u}^{(j+1)} = \underline{\underline{A}}^{-1}\underline{b}$. Note for non-periodic boundary conditions e.g. Dirichlet boundaries (fixed), $\underline{\underline{A}}$ is tridiagonal, so very easy to solve.

Advantage of Crank-Nicolson: $\mathcal{O}(\Delta x^2, \Delta t^2)$ (2nd order in space and time) and unconditionally stable.

9.4 Flux-limited diffusion (*tempered*)

As of the approximation in the diffusion equation infinitely large signal speeds (unphysical) can result. Instead of including higher order terms in the Taylor expansion ($\rightarrow$ hyperbolic system with finite information speed) we limit the speed by hand using a limited diffusion flux.

Idea: Express the flux as a product of density and velocity and limit this velocity akin to Special Relativity (based on a momentum consideration), e.g. to the speed of sound.

Let us start with the diffusion equation

$$\partial_t u = -\partial_x J, \quad J = -D\partial_x u \tag{423}$$

We make the Ansatz

$$J(x,t) = u(x,t)\cdot v, \quad textcharacteristicvelocityv \tag{424}$$

and define a *negative specific diffusion momentum*, really just the negative of the not-yet-limited velocity v.

$$-v = -\frac{J}{u} = \frac{D\partial_x u}{u} \equiv R \tag{425}$$

Now we ask ourselves: What if R was a relativistic momentum? What velocity $\tilde{v}$ would have brought forth this relativistic momentum will naturally be limited to be smaller than c. So drawing from $p = \gamma m v, v < c$ we write

$$R = -\frac{\tilde{v}}{\sqrt{1 - \frac{\tilde{v}^2}{c^2}}} \quad \leftrightarrow \quad \tilde{v} = -\frac{R}{\sqrt{1 + \frac{R^2}{c^2}}} \tag{426}$$

so we use the altered flux

$$\tilde{J} = u\tilde{v}, \quad \partial_t u = \partial_x \tilde{J} \tag{427}$$

9.5 Diffusion in three dimensions

In 3D, we have

$$\partial_t u(\underline{x}, t) = -\underline{\nabla} \cdot \underline{J}(\underline{x}, t), \quad \text{flux vector } \underline{J}(\underline{x}, t) = -\underline{\underline{D}} \cdot \underline{\nabla} u(\underline{x}, t), \quad \text{diffusion matrix } \underline{\underline{D}} \tag{428}$$

For a magnetized plasma, movement is mostly along the magnetic field lines (diffusion along the lines is dampened by collisions, diffusion perpendicular can only exist as of collisions).

$$\begin{gathered} \underline{B} = B\underline{\hat{e}}_Z \to D = \begin{pmatrix} D_\perp & 0 & 0 \\ 0 & D_\perp & 0 \\ 0 & 0 & D_\parallel \end{pmatrix} \\ \text{typically in MHD: } D_\perp \ll D_\parallel \to \partial_t \underline{u}(\underline{x}, t) = \underline{\nabla} \cdot D_\parallel \underline{\hat{e}}_z \left(\underline{\hat{e}}_z \underline{\nabla} \underline{u}(\underline{x}, t) \right) \end{gathered} \tag{429}$$

10 Solving Linear Equations with Iterative Solvers and the Multigrid Technique

Relevance of solving linear equations for simulations: Implicit schemes lead to linear systems of equations which we have to solve, examples already covered are implicit Euler applied to a linear ODE in sec. 3.4.5, or to a non-linear one (the linear equation then follows from doing the implicit step by Newton-Raphson where the Jacobian is to be inverted (sec. 3.4.6)), or just recently in the diffusion equation.

Another example of a linear system we care about is solving a discretized Poisson equation.

> Direct inversion or LU or QU decomposition of our linear system (the matrix describing it) are often too costly, so iterative methods - methods where we construct a sequence of approximate solutions that hopefully converge to the exact solution - are used.

> In the multigrid-technique, differential equations are solved using a hierarchy of discretizations.

10.1 Motivational Example 1: From the Poisson equation we can get a possibly big linear system

Let us discretize the 1D Poisson equation

$$\begin{gathered}\partial_x^2\phi = 4\pi G\rho \rightarrow (\partial_x^2\phi) \approxeq \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{h^2} \approxeq 4\pi G\rho_i \\ i = 1, \ldots, N, \quad \text{spacing } h\end{gathered} \tag{430}$$

We write this as a matrix equation

$$\underline{\underline{A}}\underline{\phi} = \underline{b}, \quad \underline{\phi} = \begin{pmatrix}\phi_1 \\ \phi_2 \\ \vdots \\ \phi_N\end{pmatrix}, \quad \underline{b} = 4\pi G\underline{\rho}, \underline{\rho} = \begin{pmatrix}\rho_1 \\ \rho_2 \\ \vdots \\ \rho_N\end{pmatrix} \tag{431}$$

with in the case of periodic boundary conditions

$$\underline{\underline{A}} = \frac{1}{h^2}\begin{pmatrix} -2 & 1 & 0 & \dots & 0 & 1 \\ 1 & -2 & 1 & 0 & \dots & \\ 0 & 1 & -2 & 1 & 0 & \dots \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & & 1 & -2 & 1 \\ 1 & 0 & 0 & \dots & 1 & -2 \end{pmatrix} \tag{432}$$

Problem: LU-decomposition or Gauss elimination with pivoting are generally $\mathcal{O}(N^3)$ (for such a sparse matrix we can do better though). Imagine a discretization in 3D, then for 1000 grid points per dimension $\underline{\underline{A}}$ would be $10^9 \times 10^9$ (mind the discretization in 3D is not *as* simple as in 1D).

10.2 Poisson equation and solving a tridiagonal system

10.2.1 1D heat Diffusion equation with Dirichlet boundaries in matrix form

Consider simple heat diffusion with constant heating rate ϵ.

$$-D\partial_x^2 T = \epsilon \tag{433}$$

so a 1D Poisson equation with constant source term ϵ.

Discretized we get as before

$$\frac{T_{i+1} - 2T_i + T_{i-1}}{\delta^2} \approxeq -\frac{\epsilon}{D} \tag{434}$$

So for a state vector

$$\underline{T} = \begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ T_N \end{pmatrix} \tag{435}$$

with Dirichlet boundary conditions $T_1 = T_N := T_0$ we get the matrix equation

$$\underline{\underline{A}}\,\underline{T} = \underline{b}, \quad \underline{b} = \begin{pmatrix} T_0 \\ -\frac{\delta^2 \epsilon}{D} \\ \vdots \\ -\frac{\delta^2 \epsilon}{D} \\ T_0 \end{pmatrix}, \quad \underline{\underline{A}} = \frac{1}{h^2} \begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & -2 & 1 & 0 & \dots & \\ 0 & 1 & -2 & 1 & 0 & \dots \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & & 1 & -2 & 1 \\ 0 & 0 & 0 & \dots & 0 & 1 \end{pmatrix} \tag{436}$$

So for a 1D Poisson problem with Dirichlet (constant) boundary conditions we get a tridiagonal matrix.

10.2.2 Forward elimination backward substitution method for solving a tridiagonal system

Consider the general tridiagonal matrix

$$\underline{\underline{A}} = \begin{pmatrix} d_1 & u_1 & 0 & \dots & 0 & 0 \\ l_1 & d_2 & u_2 & 0 & \dots & \\ 0 & l_2 & d_3 & u_3 & 0 & \dots \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & & l_{N-2} & d_{N-1} & u_{N-1} \\ 0 & 0 & \dots & 0 & l_{N-1} & d_N \end{pmatrix} \tag{437}$$

where we want to solve

$$\underline{\underline{A}}\,\underline{x} = \underline{b} \tag{438}$$

for $\underline{x}$. We use elementary operations $\underline{\underline{E}}_i$ to add multiples of rows in $\underline{\underline{A}}$ to other rows in $\underline{\underline{A}}$.

1. (Forward elimination) Using those operations, we succesively (top-down) eliminate the lower diagonal $\underline{l}$ of $\underline{\underline{A}}$.

$$\underline{\underline{E}}_1 \underline{\underline{E}}_2 \cdots \underline{\underline{E}}_{N-1} \underline{\underline{A}}\,\underline{x} = \begin{pmatrix} d_1 & u_1 & 0 & \dots & 0 & 0 \\ 0 & \tilde{d}_2 & u_2 & 0 & \dots & \\ 0 & 0 & \tilde{d}_3 & u_3 & 0 & \dots \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & & 0 & \tilde{d}_{N-1} & u_{N-1} \\ 0 & 0 & \dots & 0 & 0 & \tilde{d}_N \end{pmatrix} \underline{x} = \underline{\underline{E}}_1 \underline{\underline{E}}_2 \cdots \underline{\underline{E}}_{N-1} \underline{b} \tag{439}$$

with $\tilde{d}_2 = d_2 - \frac{l_1}{d_1} u_1$ and $\tilde{d}_i = d_i - \frac{l_{i-1}}{\tilde{d}_{i-1}} u_{i-1}$ for $i = 3, \dots, N$.

2. (Backward substitution) We can now solve the system starting from the last row.

10.3 Classical Exact Solution: LU-decomposition*

Why is it not a good idea to numerically invert the matrix?: While $\underline{\underline{A}} \in \mathbb{R}^{N\times N}$ might often be sparse (e.g. a sparse Jacobian), $\underline{\underline{A}}^{-1}$ is dense in general and our storage might not be able to handle N^2 terms of a dense inverse. This is not a problem of stability - the inverse can be computed as numerically stable as other methods are. Matrix inversion can even by done in $\mathcal{O}(N^{\log_2 7})$ (Strassen algorithm, *Gaussian elimination is not optimal*).

Let us discuss LU-decomposition (also called LR-decomposition).

Consider $\underline{\underline{L}}, \underline{\underline{R}} \in \mathbb{R}^{N\times N}$ and $\underline{\underline{A}} \in \mathbb{R}^{N\times N}$ invertible with

$$\underline{\underline{A}} = \underline{\underline{L}}\,\underline{\underline{R}}, \quad \text{lower-left triangular matrix } \underline{\underline{L}}, \text{ upper-right triangular matrix } \underline{\underline{R}} \tag{440}$$

Note: More generally, $\underline{\underline{P}}\,\underline{\underline{A}} = \underline{\underline{L}}\,\underline{\underline{R}}$ with $\underline{\underline{P}}$ a permutation matrix is possible for every invertible matrix $\underline{\underline{A}}$. Otherwise the decomposition is not always possible, e.g. for $a_{11} = 0 \to l_{11}$ or $l_{11} = 0$, so $\underline{\underline{L}}$ or $\underline{\underline{R}}$ would be singular (not full rank), so $\underline{\underline{A}}$ would not be invertible (contradiction). So $\underline{\underline{P}}$ is used to switch rows of $\underline{\underline{A}}$.

10.3.1 Solving the linear system for $\underline{x}$ when we already know the LU-decomposition

We can split

$$\underline{\underline{A}}\,\underline{x} = \underline{\underline{L}}\,\underline{\underline{R}}\,\underline{x} \quad \leftrightarrow \quad \underline{\underline{L}}\,\underline{y} = \underline{b}, \quad \underline{\underline{R}}\,\underline{x} = \underline{y} \tag{441}$$

so into solving two triangular systems.

1. (Forward substitution) Solve $\underline{\underline{L}}\,\underline{y} = \underline{b}$ for $\underline{y}$. We can write the system as

$$\begin{pmatrix} l_{11} & \\ \underline{L}_{*1} & \underline{\underline{L}}_{**} \end{pmatrix} \begin{pmatrix} y_1 \\ \underline{y}_* \end{pmatrix} = \begin{pmatrix} b_1 \\ \underline{b}_* \end{pmatrix} \Rightarrow l_{11}y_1 = b_1, \quad \underline{L}_{*1}y_1 + \underline{\underline{L}}_{**}\underline{y}_* = b_* \tag{442}$$

from which we can solve for $y_1 = \frac{b_1}{l_{11}}$. We can then construct a new triangular system for $\underline{y}_*$

$$\underline{\underline{L}}_{**}\underline{y}_* = \underline{b}_* - \underline{L}_{*1}y_1 \tag{443}$$

so $\underline{y}$ can be solved recursively, yielding the explicit formula

$$y_i = \frac{1}{l_{ii}}\left(b_i - \sum_{k=1}^{i-1} l_{ik}y_k\right) \tag{444}$$

taking $\mathcal{O}(N^2)$ operations.

2. (Backward substitution) Then $\underline{\underline{R}}\underline{x} = \underline{y}$ for $\underline{x}$. The logic is the same as for forward substitution, but bottom-up. We can write the system as

$$\begin{pmatrix} \underline{\underline{R}}_{**} & \underline{R}_{*n} \\ & r_{nn} \end{pmatrix} \begin{pmatrix} \underline{x}_* \\ x_1 \end{pmatrix} = \begin{pmatrix} \underline{y}_* \\ y_n \end{pmatrix} \tag{445}$$

yielding the explicit formula

$$a_i = \frac{1}{r_{ii}} \left(y_i - \sum_{k=i+1}^{N} r_{ik} a_k \right) \tag{446}$$

also taking $\mathcal{O}(N^2)$ operations.

10.3.2 Calculating the LU-decomposition in $\mathcal{O}(N^3)$

From

$$\begin{pmatrix} a_{11} & \underline{A}_{1*}^T \\ \underline{A}_{*1} & \underline{\underline{A}}_{**} \end{pmatrix} = \begin{pmatrix} l_{11} & \\ \underline{L}_{*1} & \underline{\underline{L}}_{**} \end{pmatrix} \begin{pmatrix} r_{11} & \underline{R}_{1*}^T \\ & \underline{\underline{R}}_{**} \end{pmatrix}, \quad \text{set diagonal of } \underline{\underline{L}} \text{ to } 1 \rightarrow \text{ unique decomp} \tag{447}$$

we can devise an $\mathcal{O}(N^3)$ algorithm to calculate $\underline{\underline{L}}$ and $\underline{\underline{R}}$, see code 2.

```julia
# in-place decomposition of A into L and R, the upper-right part of A is overwritten by R,
# the lower-left part of A is overwritten by L without diagonal (set to 1)
function lu_decomposition!(A::Matrix{T}) where T <: Number
    N = size(A, 1)
    for i in 1:N
        for j in i+1:N
            A[j, i] /= A[i, i]
            for k in i+1:N
                A[j, k] -= A[j, i] * A[i, k]
            end
        end
    end
end
# one can then extract L = tril(A, -1) + I
# and U = triu(A) via the LinearAlgebra package
```

Code-Snippet 2: LU-decomposition of a matrix $\underline{\underline{A}} \in \mathbb{R}^{N\times N}$ in $\mathcal{O}(N^3)$.

Note: Blocked versions can be parallelized, QR-decomposition retains the condition number of $\underline{\underline{A}}$.

Problem: LU-decomposition is $\mathcal{O}(N^3)$.

10.4 Jacobi iteration | a splitting method

Aim: We want quicker than $\mathcal{O}(N^3)$, approximate solutions to linear systems.

Consider the linear system

$$\underline{\underline{A}}\,\underline{x} = \underline{b}, \quad \underline{\underline{A}} \in \mathbb{R}^{N\times N}, \underline{x}, \underline{b} \in \mathbb{R}^N \tag{448}$$

For Jacobi iteration, we start with the trivial decomposition

$$\underline{\underline{A}} = \underline{\underline{D}} - (\underline{\underline{L}} + \underline{\underline{U}}), \quad \text{diagonal part } \underline{\underline{D}}, \quad \text{negative left-below-diagonal part} \underline{\underline{L}} \\ \text{negative right-above-diagonal part} \underline{\underline{U}} \tag{449}$$

so

$$\underline{\underline{A}}\,\underline{x} = \left[\underline{\underline{D}} - (\underline{\underline{L}} + \underline{\underline{U}})\right]\underline{x} = \underline{b} \quad \rightarrow \quad \underline{x} = \underline{\underline{D}}^{-1}\underline{b} + \underline{\underline{D}}^{-1}(\underline{\underline{L}} + \underline{\underline{U}})\underline{x} \tag{450}$$

so $\underline{x}$ is a fixed point of the RHS, leading to fixed-point, here Jacobi iteration

$$\boxed{\underline{x}^{(n+1)} = \underline{\underline{D}}^{-1}\underline{b} + \underline{\underline{D}}^{-1}(\underline{\underline{L}} + \underline{\underline{U}})\underline{x}^{(n)}, \quad (\underline{\underline{D}}^{-1})_{ii} = \frac{1}{A_{ii}}} \tag{451}$$

10.4.1 When does the Jacobi iteration converge?

The Jacobi iteration converges if and only if all eigenvalues λ_i of the convergence matrix

$$\underline{\underline{M}} := \underline{\underline{D}}^{-1}(\underline{\underline{L}} + \underline{\underline{U}}) \tag{452}$$

are smaller than one.

$$\text{spectral radius } \rho_s(\underline{\underline{M}}) \equiv \max_i |\lambda_i| < 1 \tag{453}$$

The smaller the spectral radius, the faster the convergence.

10.4.1.1 Derivation of the convergence criterion

Consider the error at step $n+1$

$$\begin{aligned} \underline{e}^{(n+1)} = \underline{x}_{\text{exact}} - \underline{x}^{(n+1)} &\underbrace{=}_{\text{Jacobi-Iteration}} \underline{x}_{\text{exact}} - \left(\underline{\underline{D}}^{-1}\underline{b} + \underline{\underline{M}}\,\underline{x}^{(n)}\right) \\ &\underbrace{=}_{\underline{x}_{\text{exact}} = \underline{\underline{D}}^{-1}\underline{b} + \underline{\underline{M}}\,\underline{x}_{\text{exact}}} \underline{\underline{M}}(\underline{x}_{\text{exact}} - \underline{x}^{(n)}) = \underline{\underline{M}}\,\underline{e}^{(n)} \end{aligned} \tag{454}$$

so

$$\underline{e}^{(n)} = \underline{\underline{M}}^{n}\underline{e}^{(0)} \tag{455}$$

which scales with $\rho_s^n(\underline{\underline{M}})$ for n sufficiently large (follows from decomposing $\underline{e}^{(0)}$ into eigenvectors of $\underline{\underline{M}}$) so the convergence criterion follows.

Problem: The convergence is usually slow, better use Gauss-Seidel iteration.

10.4.2 Example Jacobi Step

Consider

$$\underline{\underline{A}}\,\underline{x} = \begin{pmatrix} 10 & -5 & 0 \\ -5 & 10 & -5 \\ 0 & -5 & 5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \underline{b} = \begin{pmatrix} 0 \\ 0 \\ 15 \end{pmatrix} \tag{456}$$

then

$$\underline{\underline{D}} = \begin{pmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 5 \end{pmatrix}, \quad \underline{\underline{L}} = \begin{pmatrix} 0 & 0 & 0 \\ 5 & 0 & 0 \\ 0 & 5 & 0 \end{pmatrix}, \quad \underline{\underline{R}} = \begin{pmatrix} 0 & 5 & 0 \\ 0 & 0 & 5 \\ 0 & 0 & 0 \end{pmatrix} \tag{457}$$

so

$$\underline{x}^{(0)} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad \rightarrow \quad \underline{x}^{(1)} = \begin{pmatrix} \frac{1}{10}b_0 + \frac{1}{10} \cdot 5x_2^{(0)} \\ \frac{1}{10}b_1 + \frac{1}{10} \cdot \left(5x_1^{(0)} + 5x_3^{(0)}\right) \\ \frac{1}{5}b_1 + \frac{1}{5} \cdot 5x_2^0 \end{pmatrix} \tag{458}$$

10.5 Gauss-Seidel iteration | better splitting method

Idea: In each step $\underline{x}^{(n)} \rightarrow \underline{x}^{(n+1)}$ ($\underline{x} \in \mathbb{R}^N$) consists of N scalar updates. If we do these N scalar updates iteratively (not in parallel) we can use the results from previous updates.

10.5.1 Motivational Example

For instance in the example above

$$\underline{x}^{(0)} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad \rightarrow \quad \underline{x}^{(1)} = \begin{pmatrix} \frac{1}{10}b_0 + \frac{1}{10} \cdot 5x_2^{(0)} \\ \frac{1}{10}b_1 + \frac{1}{10} \cdot \left(5x_1^{(1)} + 5x_3^{(0)}\right) \\ \frac{1}{5}b_1 + \frac{1}{5} \cdot 5x_2^1 \end{pmatrix} \tag{459}$$

10.5.2 Gauss-Seidel update

Let us devise a different fix-point equation

$$(\underline{\underline{D}} - \underline{\underline{L}})\underline{x} = \underline{\underline{U}}\underline{x} + \underline{b} \rightarrow \underline{x} = (\underline{\underline{D}} - \underline{\underline{L}})^{-1}\underline{\underline{U}}\underline{x} + (\underline{\underline{D}} - \underline{\underline{L}})^{-1}\underline{b} \tag{460}$$

from which we follow the fix-point iteration

$$\underline{x}^{(n+1)} = (\underline{\underline{D}} - \underline{\underline{L}})^{-1}\underline{\underline{U}}\underline{x}^{(n)} + (\underline{\underline{D}} - \underline{\underline{L}})^{-1}\underline{b} \tag{461}$$

Problem: We cannot easily compute $(\underline{\underline{D}} - \underline{\underline{L}})^{-1}$.

Therefore, multiply with $(\underline{\underline{D}} - \underline{\underline{L}})$.

$$(\underline{\underline{D}} - \underline{\underline{L}})\underline{x}^{(n+1)} = \underline{\underline{U}}\underline{x}^{(n)} + \underline{b} \quad \rightarrow \quad \underline{x}^{(n+1)} = \underline{\underline{D}}^{-1}\underline{\underline{L}}\underline{x}^{(n+1)} + \underline{\underline{D}}^{-1}\underline{\underline{U}}\underline{x}^{(n)} + \underline{\underline{D}}^{-1}\underline{b} \tag{462}$$

But isn't this implicit now with $\underline{x}^{(n+1)}$ on both sides?: $\underline{\underline{L}}$ is a lower diagonal matrix (diagonal and everything above zero), so in $\underline{\underline{L}}\underline{x}^{(n+1)}$ we always only need values we have already calculated if we solve consecutively ($\rightarrow$ problem for parallelization). This is illustrated in fig. 77.

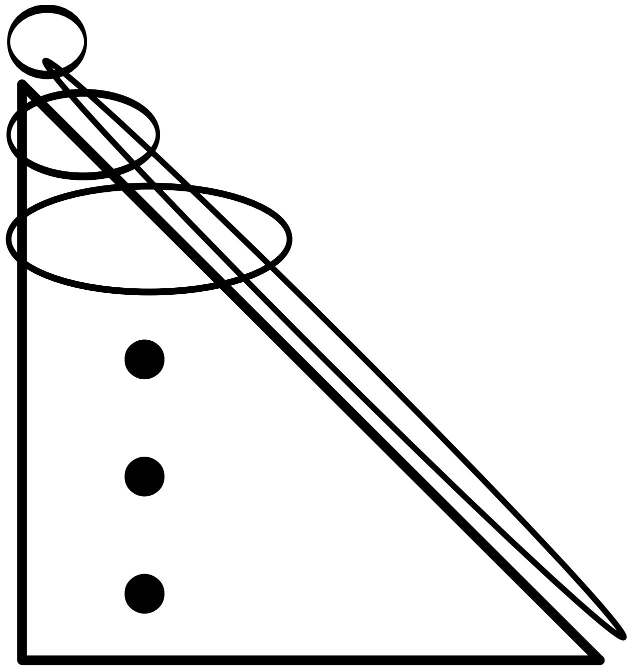


Figure 77: As of $\underline{\underline{L}}\underline{x}^{(n+1)}$ to calculate $x_i^{(n+1)}$ we only need $x_0^{(n+1)}, \ldots, x_{i-1}^{(n+1)}$ which we have already calculated.

Advantage of Gauss-Seidel: Convergence is sped up (often by a factor of 2) compared to Jacobi iteration. Convergence is guaranteed if $\underline{\underline{A}}$ is strictly diagonally dominant $|A_{ii}| > \sum_{j \neq i} |A_{ij}| \quad \forall i$ or symmetric and positive definite.

10.5.3 The problem of parallelization in Gauss-Seidel and red-black ordering

Problem: In Gauss-Seidel, the equations must be solved in sequential order - this cannot be parallelized. A general problem of using information from the same step is that the overall result is order dependent (which element is selected first).

Idea: Do not do a scheme with sequential dependence (where information is used as soon as available) but if possible e.g. rather split the N updates into two groups where all updates within a group are independent, but the updates of the second group depend on the ones from first.

For instance for the 2d-Poisson equation, *a red-black ordering* scheme can be derived, see figure 78 (details follow).

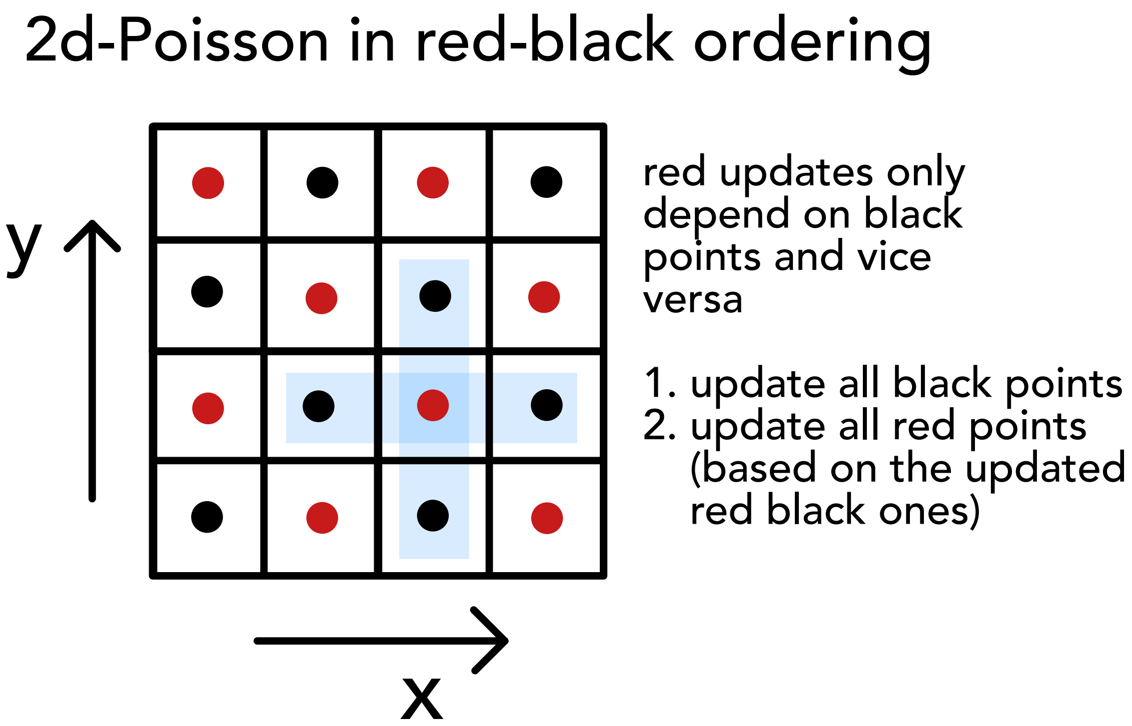


Figure 78: Red-black ordering for the 2D Poisson equation.

10.6 Relaxation problem | Poisson equation in red-black ordering

We can physically understand why applying an iterative scheme to a stationary (elliptic) problem makes sense in terms of relaxation of a dynamical system.

10.6.1 Formulating an elliptic equation as an equilibrium of a relaxation problem

Consider the elliptic equation

$$L\underline{x} = \underline{b} \tag{463}$$

for instance for the gravitational 2D-Poisson problem $(\partial_x^2 + \partial_y^2)\phi = 4\pi G\rho$.

The elliptic problem can be written as the equilibrium of the relaxation problem

$$\frac{1}{K}\partial_t\underline{x} = L\underline{x} - \underline{b}, \quad \text{for } \partial_t\underline{x} = 0 \text{ for } t \to \infty, \quad \text{some factor } K in \frac{\mathrm{m}^2}{s} \tag{464}$$

10.6.2 Red-black ordering for the 2D Poisson equation

Consider for instance the 2D Poisson equation, formulated as a relaxation problem

$$K\frac{\phi_{i,j}^{(n+1)} - \phi_{i,j}^{(n)}}{\Delta t} = \frac{\phi_{i+1,j}^{(n)} - 2\phi_{i,j}^{(n)} + \phi_{i-1,j}^{(n)}}{\Delta x^2} + \frac{\phi_{i,j+1}^{(n)} - 2\phi_{i,j}^{(n)} + \phi_{i,j-1}^{(n)}}{\Delta x^2} - 4\pi G\rho_{ij} \tag{465}$$

so

$$\phi_{i,j}^{(n+1)} = \frac{K\Delta t}{\Delta x^2}\left(\phi_{i,j}^{(n+1)} + \phi_{i-1,j}^{(n)} + \phi_{i,j+1}^{(n)} + \phi_{i,j-1}^{(n)}\right) + \underbrace{\left(1 - 4\frac{K\Delta t}{\Delta x^2}\right)}_{=0 \text{by choice } \Delta t = \frac{1}{4}\frac{K}{\Delta x^2}} \phi_{i,j}^{(n)} - 4\pi G\rho_{ij}\Delta t \tag{466}$$

where the time-step choice is akin to the CFL criterion in diffusion.

The discretized Poisson-relaxation update lends itself naturally to a red-black ordering scheme.

10.7 Multigrid technique

We have now gained an understanding of iteratively solving a linear equation as a relaxation problem.

Note: In each update step information travels as given by the stencil, for the Poisson equation only between nearest neighbors. This also limits the largest possible *timestep* to the CFL criterion.

Problem: As only cells in the stencil (often only neighboring cells) communicate per step in Jacobi and Gauss-Seidel iteration, we have to make a compromise between speed of convergence and resolution:

- On a coarse grid, information travels quicker through space (in less steps) but the resolution is bad
- On a fine grid, resolution is good, but long range interactions take lots of computational steps and long-wavelength errors (in the error vector $\underline{e}$) die out only very slowly

Idea: Start on a coarse grid, where information travels quickly, long range correlations are taken care of and we have fast convergence, then get the information onto a fine grid and resolve the details.

1. But how can we map from a coarser to a finer grid and vice versa?
2. How can we solve $\underline{\underline{A}}\underline{x} = \underline{b}$ on the coarser grid?

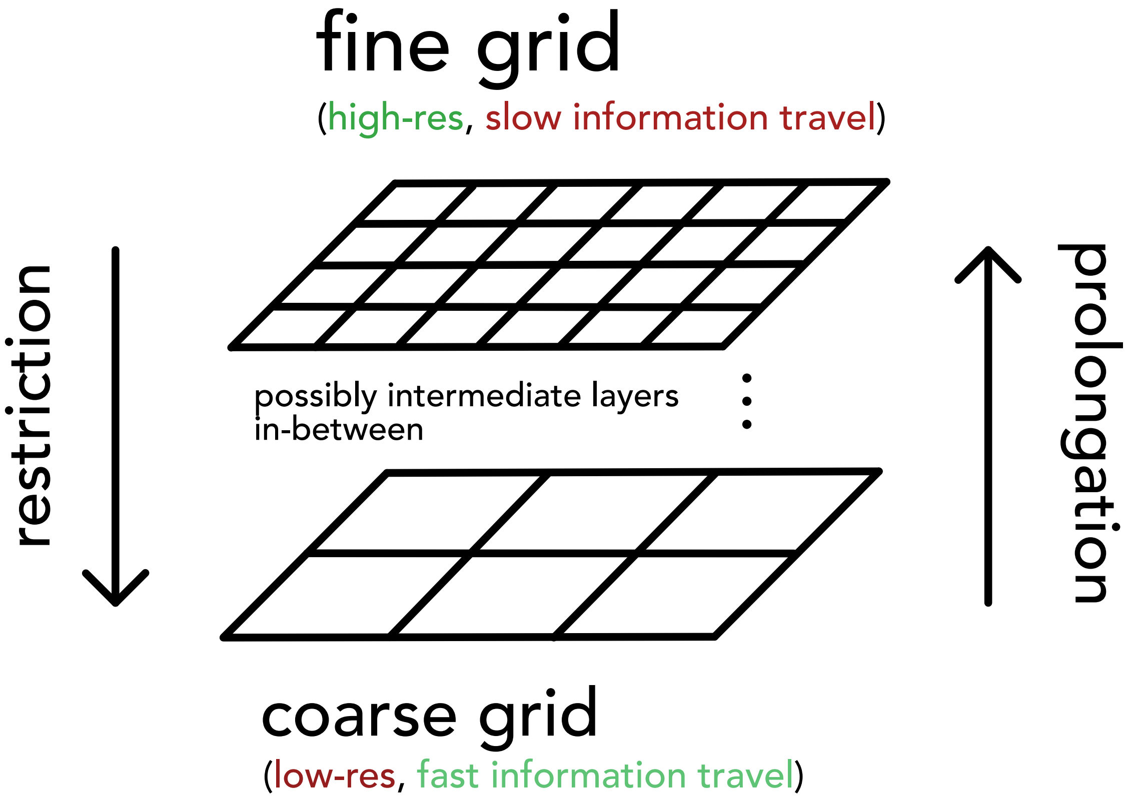


Figure 79: Grids of different resolution.

Note: Coarse-to-fine is called prolongation, fine-to-coarse restriction (see figure 79).

10.7.1 Getting finer and coarser | prolongation and restriction

Note: In the following we will only consider the case of a 1D grid (not 2D as illustrated before).

Consider meshes

1. Let $\Omega^{(h)}$ denote a 1D mesh with N cells $i = 1, \dots, N$ with spacing h.
2. Let $\Omega^{(2h)}$ denote a 1D mesh with $N/2$ cells $i = 1, \dots, N/2$ with spacing $2h$.

and let

1. $\underline{x}_i^{(h)} \in \mathbb{R}^N$ denote the solution on $\Omega^{(h)}$, so $x_i^{(h)}$ is the solution on cell i of $\Omega^{(h)}$ (e.g. a density $\rho_i^{(h)}$).
2. $\underline{x}_i^{(2h)} \in \mathbb{R}^{N/2}$ denote the solution on $\Omega^{(2h)}$, so $x_i^{(2h)}$ is the solution on cell i of $\Omega^{(2h)}$.

Restriction and prolongation in 1D are illustrated in figure 80.

Prolongation and restriction are linear operators written as matrices $\underline{\underline{I}}^{\text{to this spacing}}_{\text{from this spacing}}$.

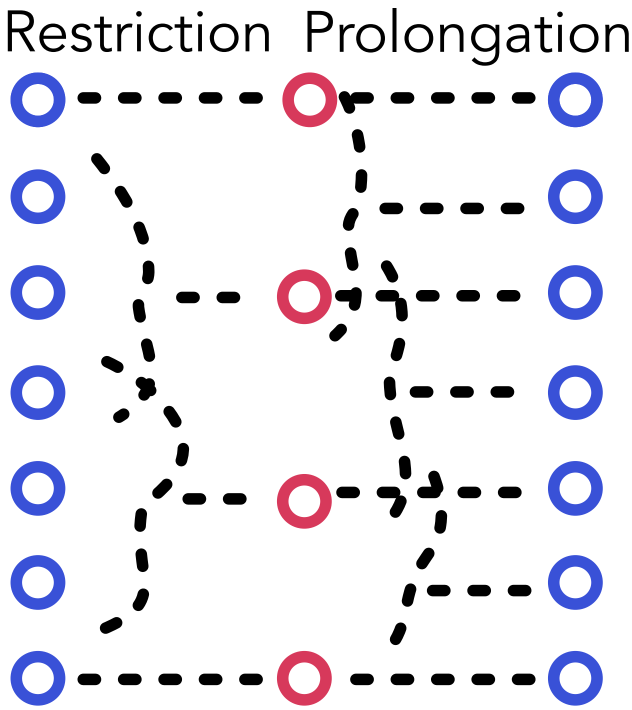


Figure 80: Illustration of restriction and prolongation.

10.7.1.1 Coarse-to-fine | interpolation called prolongation

We map from coarse to fine using

$$\underline{\underline{I}}_{2h}^{h}\underline{x}^{(2h)} = \underline{x}^{(h)} \tag{467}$$

for instance by

$$\underline{\underline{I}}_{2h}^{h} \in \mathbb{R}^{N\times\frac{N}{2}} : \text{for } 0 \leq i < \frac{N}{2} : x_{2i}^{(h)} = x_i^{(2h)}, \quad x_{2i+1}^{(h)} = \frac{1}{2}\left(x_i^{(2h)} + x_{i+1}^{(2h)}\right) \tag{468}$$

as shown in figure 80 and furthermore in figure 81.

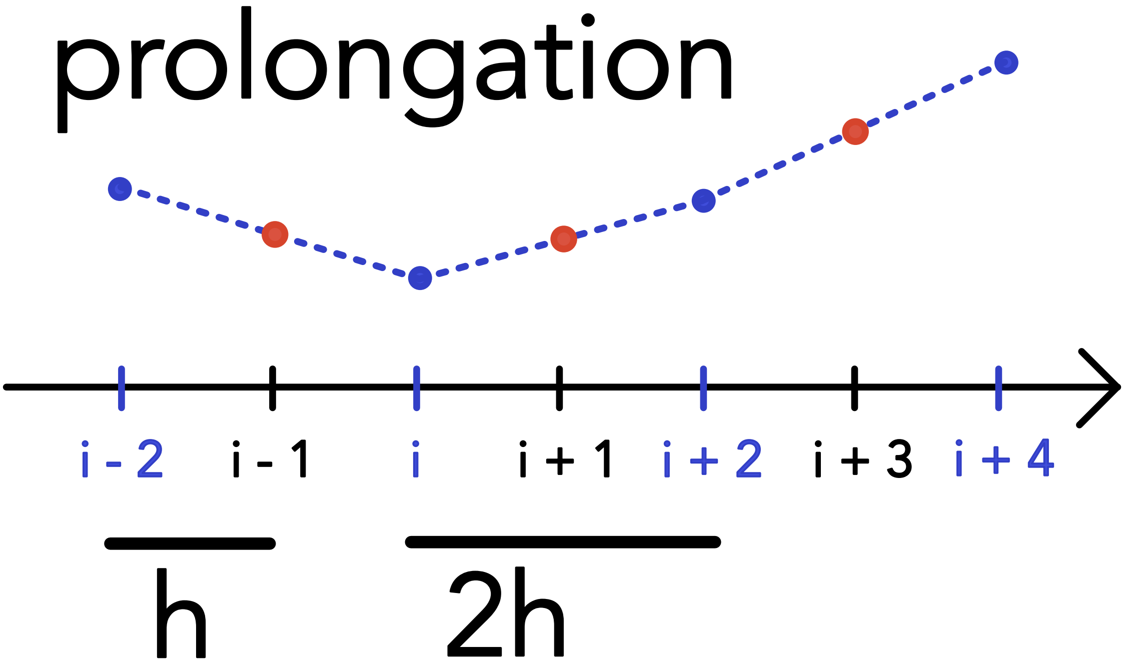


Figure 81: 1D-Prolongation from coarse to fine grid.

10.7.1.2 Fine-to-coarse | restriction

We now convert

$$\underline{\underline{I}}_h^{2h} \underline{x}^{(h)} = \underline{x}^{(2h)} \tag{469}$$

with a simple example being

$$\underline{\underline{I}}_h^{2h} \in \mathbb{R}^{\frac{N}{2} \times N} : \text{for } 0 \leq i < \frac{N}{2} : x_i^{(2h)} = \frac{x_{2i-1}^{(h)} + 2x_{2i}^{(h)} + x_{2i+1}^{(h)}}{4} \tag{470}$$

10.7.1.3 Relation of restriction and prolongation

Prolongation and restriction matrices are oftentimes constructed to be scaled transposes of each other, so

$$\underline{\underline{I}}_h^{2h} = c[\underline{\underline{I}}_{2h}^h]^T \tag{471}$$

Consider for instance

$$\begin{aligned}\text{prolongation: } \underset{=2h}{I}^{h} &= \frac{1}{2}\begin{pmatrix} 1 & & \\ 2 & & \\ 1 & 1 & \\ & 2 & \\ & 1 & 1 \\ & & 2 \\ & & 1 \end{pmatrix} \\ \text{restriction: } \underset{=h}{I}^{2h} &= \frac{1}{2}\left(\underset{=2h}{I}^{h}\right)^T = \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 & & & & \\ & & 1 & 2 & 1 & & \\ & & & & 1 & 2 & 1 \end{pmatrix}\end{aligned} \tag{472}$$

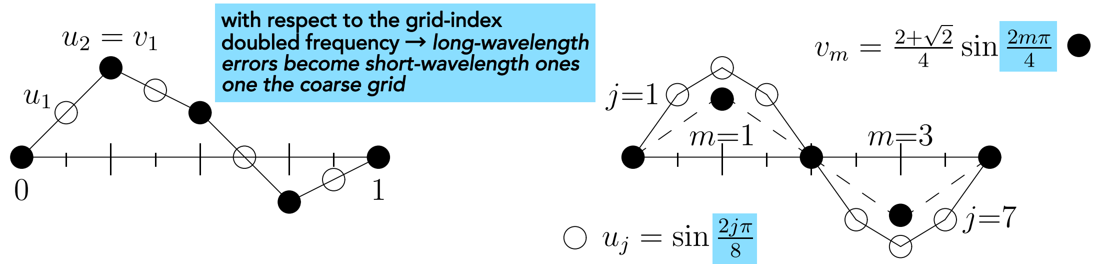
so the application of **prolongation** means

$$\frac{1}{2}\begin{pmatrix} 1 & & \\ 2 & & \\ 1 & 1 & \\ & 2 & \\ & 1 & 1 \\ & & 2 \\ & & 1 \end{pmatrix}\begin{pmatrix} x_1^{(2h)} \\ x_2^{(2h)} \\ x_3^{(2h)} \end{pmatrix} = \begin{pmatrix} x_1^{(2h)}/2 \\ x_1^{(2h)} \\ x_1^{(2h)}/2 + x_2^{(2h)}/2 \\ x_2^{(2h)} \\ x_2^{(2h)}/2 + x_3^{(2h)}/2 \\ x_3^{(2h)} \\ x_3^{(2h)}/2 \end{pmatrix} = \begin{pmatrix} x_1^{(h)} \\ x_2^{(h)} \\ x_3^{(h)} \\ x_4^{(h)} \\ x_5^{(h)} \\ x_6^{(h)} \\ x_7^{(h)} \end{pmatrix} \tag{473}$$

and **restriction**

$$\frac{1}{4}\begin{pmatrix} 1 & 2 & 1 & & & & \\ & & 1 & 2 & 1 & & \\ & & & & 1 & 2 & 1 \end{pmatrix}\begin{pmatrix} x_1^{(h)} \\ x_2^{(h)} \\ x_3^{(h)} \\ x_4^{(h)} \\ x_5^{(h)} \\ x_6^{(h)} \\ x_7^{(h)} \end{pmatrix} = \begin{pmatrix} x_1^{(2h)} \\ x_2^{(2h)} \\ x_3^{(2h)} \end{pmatrix} \tag{474}$$

Both are illustrated in figure 82, also illustrating that with respect to the grid index, low-frequency errors on the fine grid have double (so higher) frequency on the coarse grid.



(a) Linear interpolation by $u = I_{2h}^{h} v$ (b) Restriction by $R_h^{2h} u = \frac{1}{2}(I_{2h}^{h})^{\mathrm{T}} u$

Figure 82: Prolongation and restriction of a sine wave (figure by Gilbert Strang).

Note: Taking prolongation and restriction as transposes of each others breaks at the boundaries, if we do not as in figure 82 have zeroes on the boundaries.

10.7.1.4 Short-hand stencil notation

Short-hand notations for prolongation and restriction are given in table 15.

Prolongation	**Restriction**
We can write prolongation as $$\text{1D prolongation: } \underline{\underline{I}}_{2h}^{h} : \left] \ \tfrac{1}{2} \quad 1 \quad \tfrac{1}{2} \ \right[\qquad (475)$$ meaning that every coarse point is added to three fine points with these respective weights (see previous example).	We can write restriction as $$\text{1D restriction: } \underline{\underline{I}}_{h}^{2h} : \left[\ \tfrac{1}{4} \quad \tfrac{1}{2} \quad \tfrac{1}{4} \ \right] \qquad (476)$$ - the coarse point is the weighted sum of the three fine points.

Table 15: Short-hand notations for prolongation and restriction.

Similar short-hand notations can be devised for 2D and 3D.

10.7.2 Multigrid V-cycle

Our aim is still to solve $\underline{\underline{A}}\underline{x} = \underline{b}$ smartly, by - in the view of a relaxation problem - doing the rough large-scale work quickly on a coarse and the fine details on a fine grid afterwards. We still have to anwer

- How can such a procedure look like?
- How do we convert $\underline{\underline{A}}$ to coarser grids?

10.7.2.1 Initial definitions

The error is defined as

$$\underline{e} \equiv \underline{x}_{\text{exact}} - \underline{\tilde{x}}, \quad \text{current approximate solution } \underline{\tilde{x}} \tag{477}$$

Note: The residual is the error in the solution, not $\underline{x}$. The error and residual are related linearly by $\underline{\underline{A}}$.

$$\underline{\underline{A}}\underline{x}_{\text{exact}} = \underline{b} \quad \rightarrow \quad \underline{\underline{A}}(\underline{e} + \underline{\tilde{x}}) = \underline{b} \quad \rightarrow \quad \underline{\underline{A}}\underline{e} = \underline{r} \tag{478}$$

Idea: Consider we have an initial guess $\underline{\tilde{x}}$ on the fine grid. We can easily calculate the residual $\underline{r}$. We can then restrict $\underline{r}$ to a coarse grid, solve for the error there ($\underline{\underline{A}}\underline{e} = \underline{r}$) (given we can transform $\underline{\underline{A}}$ to the coarse grid) and then prolong the error to the fine grid and make the correction $\underline{x}_{\text{exact}} = \underline{\tilde{x}} + \underline{e}$.

Let us formalize this idea.

10.7.2.2 Coarse-grid correction scheme

We use the error calculated on the coarse grid to correct on the fine grid.

Let us start with a (fine-grid) guess $\underline{\tilde{x}}^{(h)}$ for the problem

$$\underline{\underline{A}}^{(h)}\underline{x}^{(h)} = \underline{b}^{(h)} \tag{479}$$

and we want to obtain an improvement $\underline{\tilde{x}}'^{(h)} = CG(\underline{\tilde{x}}^{(h)}, \underline{b}^{(h)})$.

CG consists of the following steps

1. Perform one iterative *relaxation* step on the current grid h, e.g. Jacobi or Gauss-Seidel
2. Compute the residual
$$\underline{r}^{(h)} = \underline{b} - \underline{\underline{A}}\underline{\tilde{x}}^{(h)} \tag{480}$$
3. Restrict the residual to the coarser mesh
$$\underline{r}^{(2h)} = \underline{\underline{I}}_h^{2h}\underline{r}^{(h)} \tag{481}$$
4. Solve for the error on the coarser mesh
$$\underline{\underline{A}}^{(2h)}\underline{e}^{(2h)} = \underline{r}^{(2h)}, \quad \text{starting guess } \underline{\tilde{e}}^{(2h)} = \underline{0} \tag{482}$$

5. Correct $\underline{\tilde{x}}^{(h)}$ on the finer mesh using the prolonged error $\underline{e}^{(h)} = \underline{\underline{I}}_{2h}^{h}\underline{e}^{2h}$

$$\underline{\tilde{x}}'^{(h)} = \underline{\tilde{x}}^{(h)} + \underline{e}^{(h)} \tag{483}$$

6. Further iterative *relaxation* step on the fine mesh

How to solve for the error on the coarse mesh?: Step 4 can be performed by recursively calling $CG(\underline{\tilde{x}}^{(2^i h)}, \underline{b}^{(2^i h)}), i \geq 1$ until a coarseness is reached where one can easily exactly solve or solve by multiple relaxation steps.

How to find $\underline{\underline{A}}^{(2h)}$ on the coarse mesh?: Generally in the **Galerkin coarse grid approximation** one defines

$$\underline{\underline{A}}^{(2h)} = \underline{\underline{I}}_{h}^{2h}\underline{\underline{A}}^{(h)}\underline{\underline{I}}_{2h}^{h} \tag{484}$$

which is an additional matrix operation that might enlarge the stencil (and so the computational cost), depending on the interpolation operator.

When $\underline{\underline{A}}$ was brought forth by formulating a discretization in matrix form, we can use the same discrete equations as one the fine grid (**direct method**). In other words, the same stencil (*Schablone*) is used to go over the points and collect for the linear relations, e.g. for the 2D-Poisson - no matter the coarseness - we would construct $\underline{\underline{A}}$ based on eq. 466.

10.7.2.3 V-cycle

The 6-step iteration process with a recursion call in step 4 leads to a cycle as illustrated in figure 83.

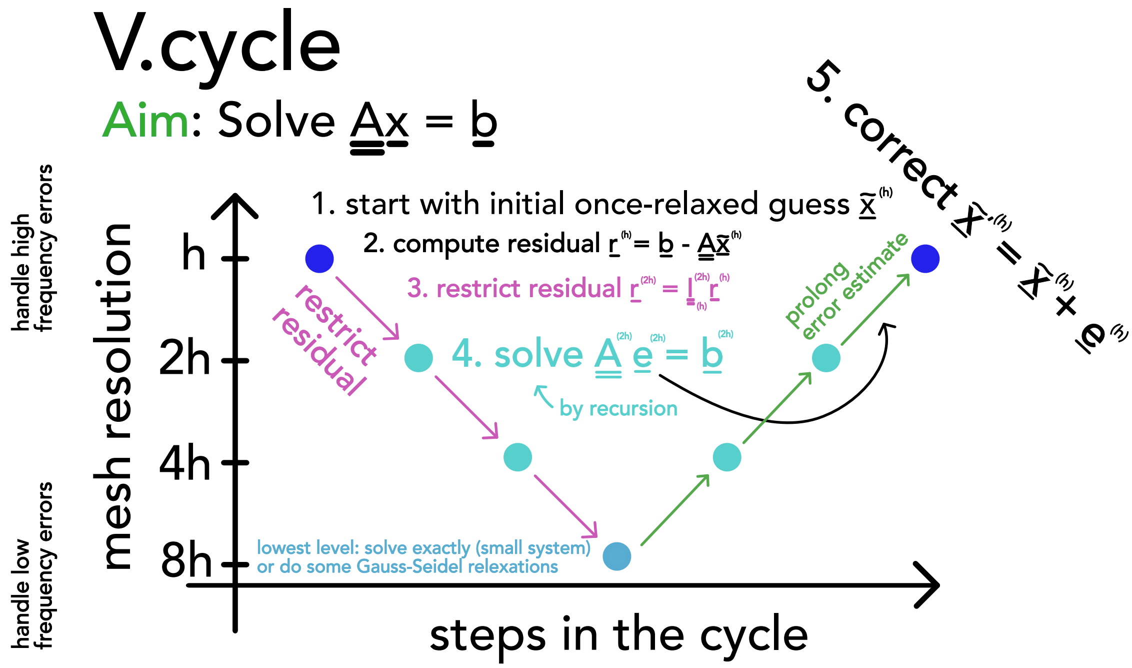


Figure 83: V-cycle.

The V-cycle for the Poisson equation has the same cost as FFT-based methods but requires less thread communication when parallelized.

Computational costs

$$\begin{gathered}\text{computational cost per V-cycle:}\mathcal{O}(N_{\text{grid}})\\ \text{computational cost until convergence to machine error (mult. cycles) } \mathcal{O}(N_{\text{grid}}\log N_{\text{grid}})\\ \text{with } N_{\text{grid}} \text{ being the number of grid-cells on the fine grid}\end{gathered} \tag{485}$$

10.7.2.4 Full multigrid method

Problem: How to make a good initial guess $\underset{\sim}{\tilde{x}}^{(h)}$ (if not available e.g. by taking the result from the last time-step in a simulation)?

Idea: First solve on the coarsest grid, then interpolate up to get a good initial guess. Based on this idea, the full multigrid cycle is constructed.

The multigrid cycle is illustrated in figure 84.

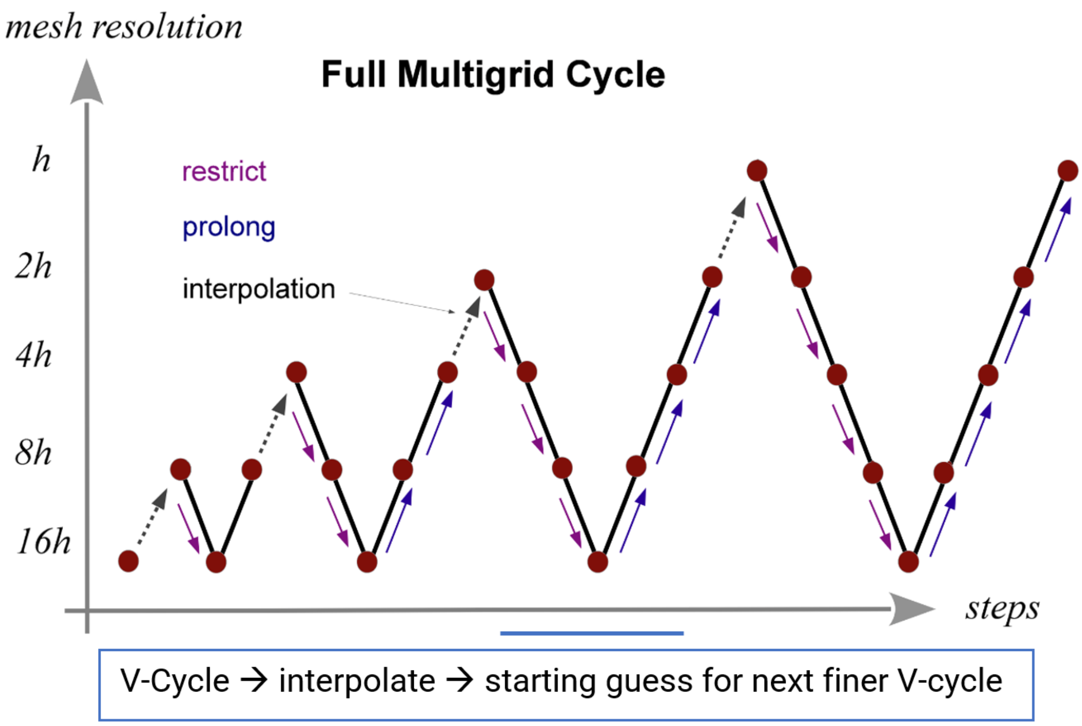


Figure 84: Full multigrid cycle.

The steps are

1. Initialize the righthand side on all grid levels $\underline{b}^{(h)}, \underline{b}^{(2h)}, \underline{b}^{(H)}$.
2. Solve $A^{(H)}\underline{x}^{(H)} = \underline{b}^{(H)}$ exactly on the coarsest level H.
3. Obtain an initial guess for a finer grid by interpolating the solution from the next coarser level below, i.e. $\underline{\tilde{x}}^{(h)} = \underline{\underline{I}}_{2h}^{h}\underline{\tilde{x}}^{(2h)}$
4. Solve the problem in the finer level using this starting guess $\underline{\tilde{x}}^{(h)}$ and one v-cycle.
5. repeat step 3 until finest level is reached

Computational cost of one multigrid cycle: As the V-cycle it has $\mathcal{O}(N_{\text{grid}})$ as N_{grid} is large compared to the cost of the V-cycle hierarchy.

10.8 Krylow subspace methods

Our aim still is to solve a linear system $\underline{\underline{A}}\underline{x} = \underline{b}$.

Problem: LU-decomposition has cubic runtime ($\sim \frac{2}{3}n^3$) and the previously discussed iterative methods might also not always be optimal.

10.8.1 Motivation for the need of Krylow subspace methods in the context of non-linear root-finding of big systems*

Consider a non-linear equation

$$\underline{0} = g(\underline{\xi}) \tag{486}$$

which we want to solve for ξ.[19] A common root-finding approach is Newton-Iteration[20]

$$\underline{\xi}_{k+1} = \xi_k - \underline{\underline{J}}_g^{-1}(\underline{\xi}_k)g(\underline{\xi}_k), \quad \text{Jacobian } \underline{\underline{J}}_g \tag{488}$$

where at the lowest level we have to solve the linear equation

$$\underline{b} := \underline{g}(\underline{\xi}_k) = \underline{\underline{J}}_g \left(\underline{\xi}_k - \underline{\xi}_{k+1}\right) = \underline{\underline{J}}_g \underline{a} \tag{489}$$

for $\underline{a}$.

Problem: But what if the resulting Jacobian is too large to be stored?

Idea: A directional derivative (a Jacobian vector product[a]) can be calculated in one (forward) automatic differentiation pass - cheaply and storage efficient. So what if we could solve $\underline{\underline{J}}\underline{a} = \underline{b}$ only based on Jacobian vector products?

[a]

$$\begin{gathered} \underline{\underline{J}}\underline{v} = (\underline{\nabla} \cdot \underline{g})\underline{v} = \lim_{h\to 0} \frac{g(\underline{x} + \underline{v}h) - g(\underline{x})}{h} \\ \left(\text{as } (\underline{\underline{J}}\underline{v})_i = \sum_{j=1}^{n} \left(\underline{\underline{J}}_g\right)_{ij} v_j = \sum_{j=1}^{n} \frac{\partial g_i}{\partial x_j} v_j = ((\underline{\nabla} \cdot \underline{g})\underline{v})_i\right) \end{gathered} \tag{490}$$

10.8.2 Motivation | solving a system of linear equations using a gradient method

Consider $\underline{\underline{A}} \in \mathbb{R}^{N\times N}$ symmetric ($\underline{\underline{A}}^T = \underline{\underline{A}}$) and positive definite, so $\underline{x}^T\underline{\underline{A}}\underline{x} > 0 \forall \underline{x} \in \mathbb{K}^N \setminus \{\underline{0}\}$ (only positive eigenvalues). Then the following holds

$$\underline{\underline{A}}\underline{x} = \underline{b} \leftrightarrow \text{quadratic form } f(x) := \frac{1}{2}\underline{x}^T\underline{\underline{A}}\underline{x} - \underline{x}^T\underline{b} \text{ is minimal} \tag{491}$$

[19]For instance an implicit Euler step can be formulated as such a root finding problem.

$$\begin{gathered} \text{implicit step } \underline{y}^{(n+1)} = \underline{y}^{(n)} + \Delta t \underline{f}\left(\underline{y}^{(n+1)}\right) \\ \to \quad \text{root-finding-problem } \underline{0} = \underline{y}^{(n+1)} - \underline{y}^{(n)} - \Delta t \underline{f}\left(\underline{y}^{(n+1)}\right) =: \underline{g}(\underline{\xi}) \end{gathered} \tag{487}$$

[20]Which has quadratic convergence - as the method converges on the root, the difference between the root and the approximation is squared in each step.

(intuitively as the quadratic form for a positive definite matrix if parabolic) where we can now apply standard gradient descent (**steepest descent method**)

$$\underline{x}^{(n+1)} = \underline{x}^{(n)} + \alpha^{(n)}\underline{p}^{(n)}, \quad \underline{p}^{(n)} = -\underline{\nabla} f(x^{(n)}) = -(\underline{b} - \underline{\underline{A}}\underline{x}^{(n)}) \tag{492}$$

so we take steps along the steepest descent with some *learning rate* $\alpha^{(n)}$

Note: Quadratic forms for different kinds of matrices are illustrated in figure 85. For an indifinite matrix the steepest descent method or the later-introduces conjugate gradient method will fail, because the solution is a saddle point. For the more general case of even non-symmetric matrices, GMRES can be used.

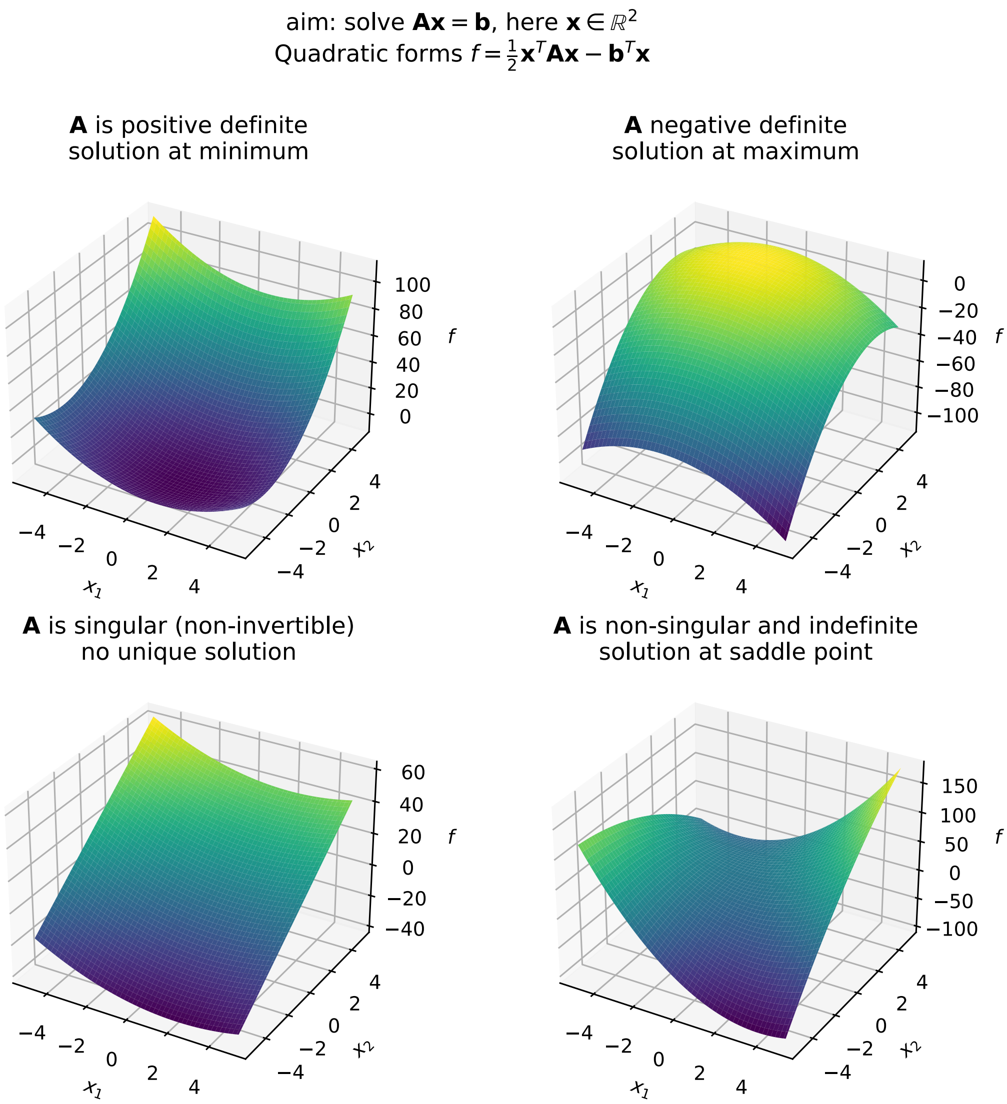


Figure 85: Quadratic forms for different kinds of matrices.

Problem: Steepest descent often takes steps in directions already taken. Steepest descent is good if our parabolic quadratic form f is nearly circular but the more elliptic f is, the more problematic steepest descent becomes, see figure 86.

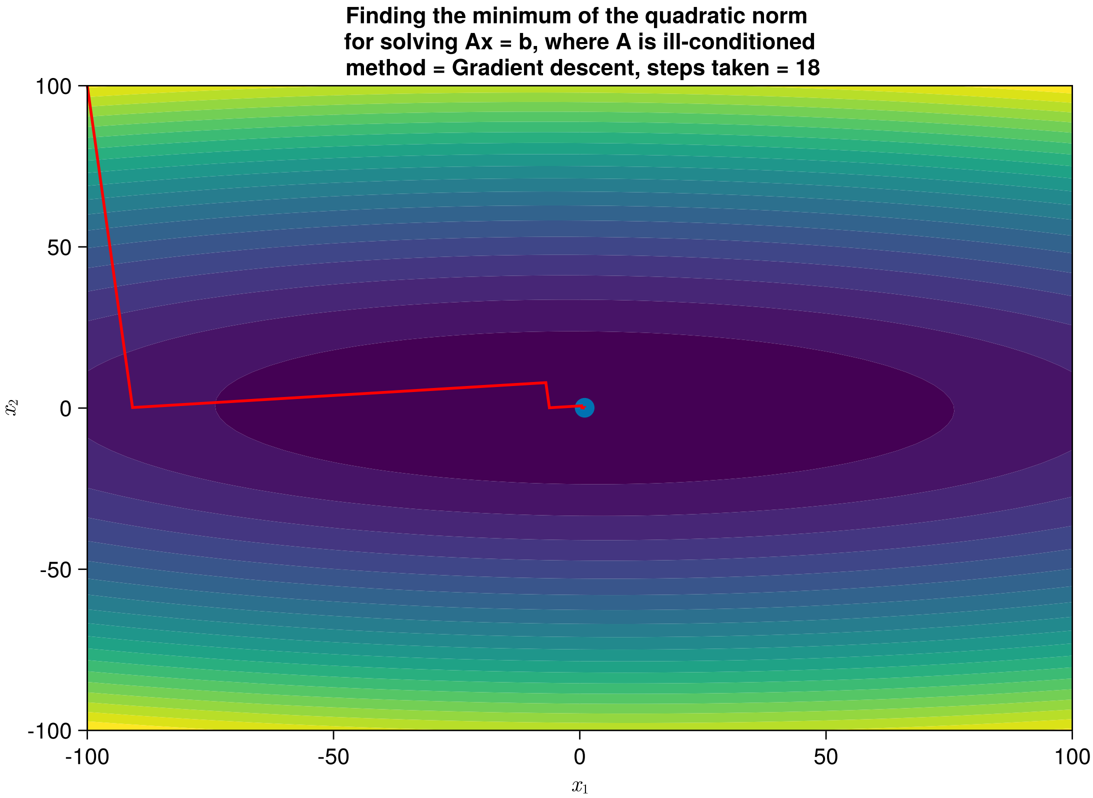


Figure 86: Steepest descent for soling a linear system $\underline{\underline{A}}\,\underline{x} = \underline{b}$ with positive definite matrix $\underline{\underline{A}}$.

Idea: Is there maybe a smarter way to choose directions $\underline{p}^{(n)}$ so that we do not go into the same direction multiple times and still find the solution?

We will introduce the conjugate gradient method, which takes at most N (size of the linear system) steps to converge to the solution, see figure 87.

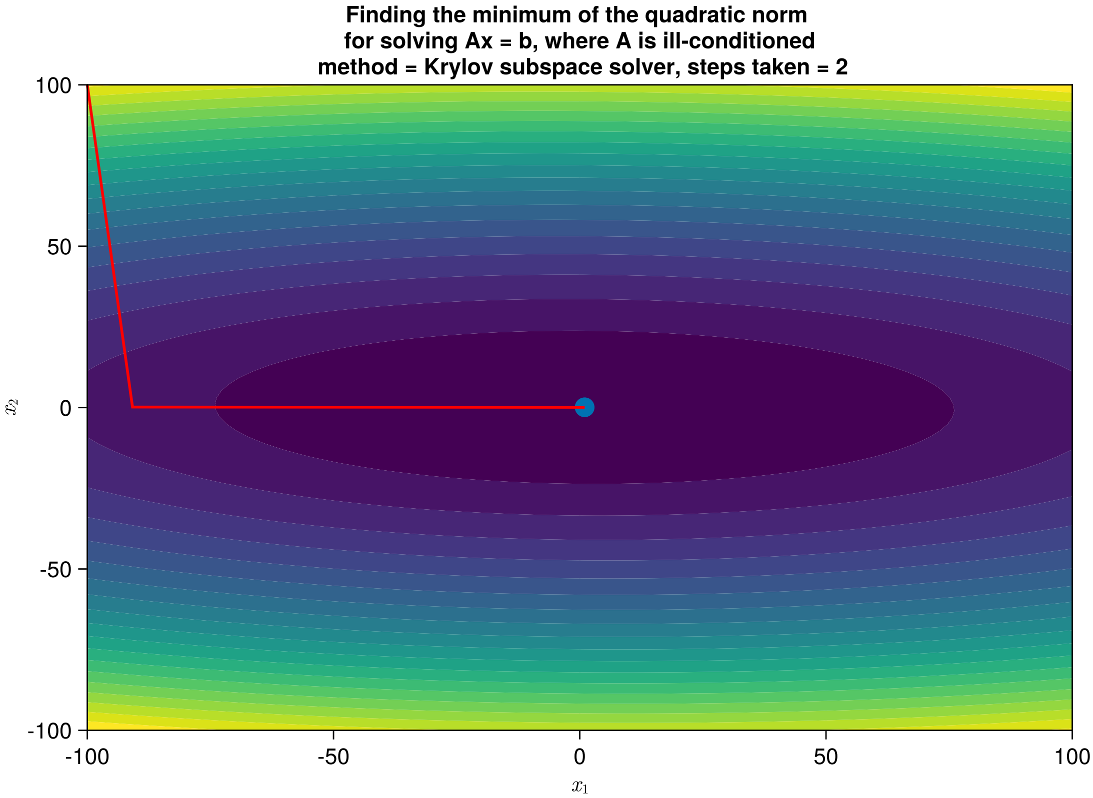


Figure 87: Conjugate gradient method for soling a linear system $\underline{\underline{A}}\,\underline{x} = \underline{b}$ with positive definite matrix $\underline{\underline{A}}$.

10.8.3 Krylov subspace and construction of iterative methods for solving $\underline{\underline{A}}\,\underline{x} = \underline{b}$

The Krylov subspace of order r is based on

$$\underline{b}, \underline{\underline{A}}\,\underline{b}, \underline{\underline{A}}^2\underline{b}, \ldots, \underline{\underline{A}}^{r-1}\underline{b} \tag{493}$$

forming a linear vector space

$$\mathcal{K}_r := \mathcal{K}_r(\underline{\underline{A}}, \underline{b}) = \mathrm{span}(\underline{b}, \underline{\underline{A}}\,\underline{b}, \underline{\underline{A}}^2\underline{b}, \ldots, \underline{\underline{A}}^{r-1}\underline{b}) \tag{494}$$

note that

- $\underline{\underline{A}}^i\underline{b}$ can be calculated by i matrix-vector products (very efficient)
- $\underline{\underline{A}}^i\underline{b}$ are linearly independent for each $i < N$, the dimension of $\mathcal{K}_r$ increases with r up to N, the dimension of $\underline{\underline{A}}$

How can we smartly choose $\underline{x}^{(n)} \in \mathcal{K}_n$ in an iterative scheme for solving $\underline{\underline{A}}\,\underline{x} = \underline{b}$?

- so that the residual $\underline{r}^{(n)} = \underline{b} - \underline{\underline{A}}\,\underline{x}^{(n)}$ is orthogonal to $\mathcal{K}_n$ (so $\in \mathcal{K}_{n+1}$) (**conjugate**

gradients)

- so that the residual has minimum norm for $\underline{x}^{(n)} \in \mathcal{K}_n$
- ...

Note: The best basis for the Krylov subspace $\mathcal{K}_r$ is orthonormal and can be computed using Arnoldi's method (essentialy Gram-Schmidt). Since $\underline{r}^{(n)} \perp \mathcal{K}_n$ it will be a scaled version of the last orthonormal base vector, one needs to get from $\mathcal{K}_n$ to $\mathcal{K}_{n+1}$

10.8.4 Conjugate gradients method

The sequence

$$\underline{x}^{(n+1)} = \underline{x}^{(n)} + \alpha^{(n)}\underline{p}^{(n)}, \quad \text{residual } \underline{r}^{(n)} = \underline{b} - \underline{\underline{A}}\underline{x}^{(n)} \tag{495}$$

with (without proof)

$$\underline{p}^{(n)} = \underline{r}^{(n)} + \frac{\underline{r}^{(n)} \cdot \underline{r}^{(n)}}{\underline{r}^{(n-1)}\underline{r}^{(n-1)}}\underline{p}^{(n-1)} \tag{496}$$

and *learning rate*[21]

$$\alpha^{(n)} = -\frac{\left(\underline{p}^{(n)}\right)^T \underline{r}^{(n)}}{\left(\underline{p}^{(n)}\right)^T \underline{\underline{A}}\underline{p}^{(n)}} \tag{497}$$

solves $\underline{\underline{A}}\underline{x} = \underline{b}$, $\underline{\underline{A}} \in \mathbb{R}^{N\times N}$, $\underline{\underline{A}}$ symmetric and positive definite, in at most N steps, so $\underline{r}^{(N)} = \underline{0}$ (here stated without proof).

The directions taken are constructed so that

- the residuals are orthogonal, $\underline{r}^{(i)} \cdot \underline{r}^{(j)} = 0$ for $i \neq j$
- the step-directions are conjugate, $\left(\underline{p}^{(j)}\right)^T \underline{\underline{A}}\underline{p}^{(j)} = 0$ for $i \neq j$
- the residuals and step-directions are mutually orthogonal, $\underline{p}^{(i)} \cdot \underline{r}^{(j)} = 0$ for $i \neq j$

[21] This follows simply from $\partial_x f|_{\underline{x}^{(n)}+\alpha^{(n)}\underline{p}^{(n)}} \underset{!}{=} 0$, where f is the quadratic form of $\underline{\underline{A}}\underline{x} = \underline{b}$.

11 Fourier methods

Derivatives in normal space turn into multiplications in Fourier space - we can converted a PDE into an algebraic equation by Fourier transform. Other applications of Fourier methods are

1. calculate correlation functions
2. projection of vector operators
3. diagonalize circulant matrices
4. image smoothing

11.1 Convolution problems | solving Poisson's equation using Fourier methods

We want to solve Poisson's equation

$$\text{gravitational } \underline{\nabla}^2\phi = 4\pi G\rho, \quad \text{electrostatics } \underline{\nabla}^2\phi = -4\pi\rho \tag{498}$$

for a given density distribution ρ (charge density in electrostatics) (here in CGS units).

Idea: The solution to the Poisson equation is generally a convolution[a], which by the convolution theorem becomes a multiplication in Fourier space - a very fast operation $\mathcal{O}(N)$. So the main cost is the Fourier transform, which can be done in $\mathcal{O}(N \log N)$.

[a] A convolution of a Greens function and the density.

11.1.0.1 The solution to the Poisson equation is a convolution

For a point source q at the origin, the density distribution is $\rho(\underline{x}) = q\delta(\underline{x})$ and we know the potential to be $\phi = \frac{q}{|\underline{x}|}$, so in the Poisson equation $\underline{\nabla}^2 \frac{1}{|\underline{x}|} = -4\pi\delta(\underline{x})$.

As of the linearity of Poisson's equation, the solution to a more complex charge distribution is

$$\phi(\underline{x}) = \sum_{i=1}^{N} \frac{q}{|\underline{x} - \underline{x}_i|} \quad \underbrace{\rightarrow}_{\text{continuous limit}} \quad \phi(\underline{x}) = \phi(\underline{x}) = \int \frac{\rho\left(\underline{x}'\right)}{|\underline{x} - \underline{x}'|} d\underline{x}' \tag{499}$$

or in the gravitational case

$$\phi(\underline{x}) = -G\int \frac{\rho\left(\underline{x}'\right)}{\left|\underline{x}-\underline{x}'\right|}d\underline{x}' = \int g\left(\underline{x}-\underline{x}'\right)\rho\left(\underline{x}'\right)d\underline{x}' = g\star\rho$$
$$\text{Greens function of gravity } g(\underline{x}) = -\frac{G}{|\underline{x}|} \tag{500}$$

- the solution for the potential is the convolution of the density and the Greens function g.

11.1.0.2 A convolution in real space turns into a multiplication in Fourier space

The Fourier transform of the convolution of two function is equal to the product of the individual Fourier transforms of those functions (**convolution theorem**).

$$\mathcal{F}(f\star g) = \mathcal{F}(f)\cdot\mathcal{F}(g), \quad \text{functions } f,g, \quad \text{Fourier transform } \mathcal{F} \tag{501}$$

We can turn a convolution in real space into a simple point-by-point multiplication in Fourier space. Applied to the Poisson problem, we get

$$\phi = \mathcal{F}^{-1}[\mathcal{F}(g)\cdot\mathcal{F}(\rho)], \quad \text{in Fourier space } \mathcal{F}(\phi) =: \hat{\phi}(\underline{k}) = \hat{g}(\underline{k})\cdot\hat{\rho}(\underline{k}) \tag{502}$$

11.1.0.3 For periodic boundaries, the Fourier transform turns from an integral to a sum

Assume our space to be a box of size L in all dimensions and the boundary conditions to be periodic.

As of the periodic boundary conditions, the set of possible k-vectors is discrete, as illustrated in figure 88.

$$\rho(\underline{x}) = \sum_{\underline{k}} \rho_{\underline{k}} \exp(i\underline{k}\cdot\underline{x})$$
$$\text{periodic boundary conditions } \rightarrow \underline{k} \in \frac{2\pi}{L}\begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix}, \quad n_1,n_2,n_3\in\mathbb{Z} \tag{503}$$
$$\text{as of } \rho\in\mathbb{R}\,(\rho^* = \rho) \rightarrow \hat{\rho}_{\underline{k}} = \hat{\rho}^*_{-\underline{k}}$$

We have an infinite but discrete grid in reciprocal space.

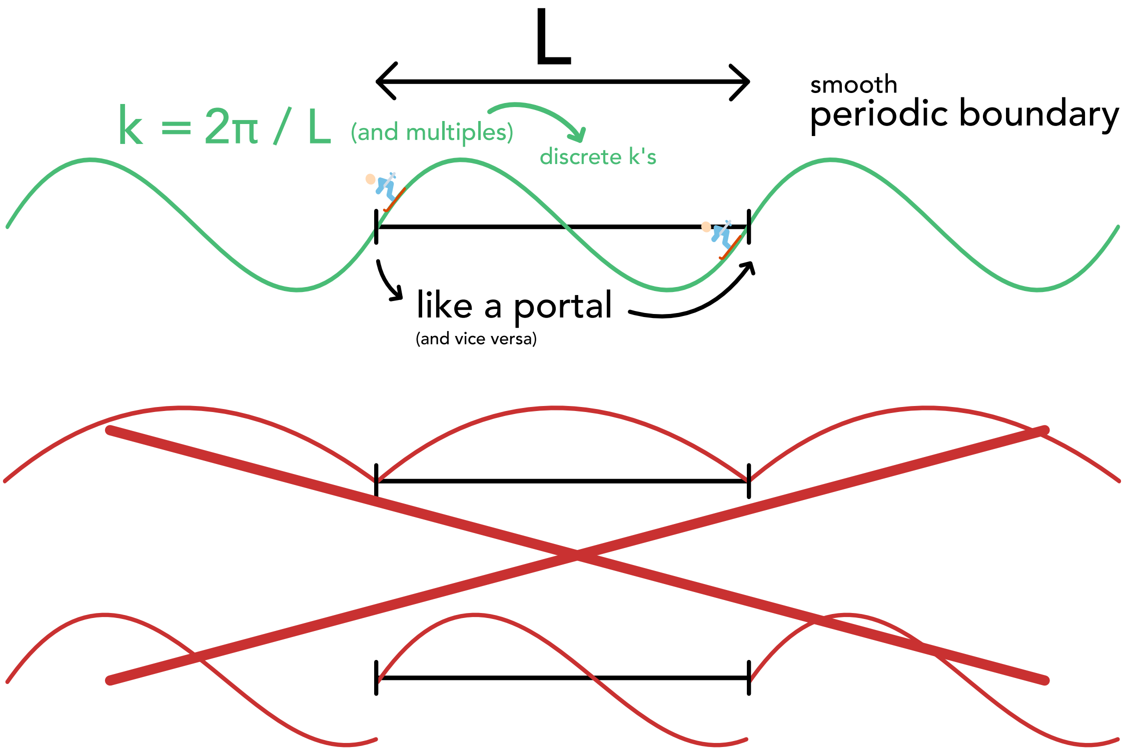


Figure 88: The set of possible k-vectors is discrete for periodic boundary conditions.

The Fourier transform over the periodic density in real space is

$$\hat{\rho}_{\underline{k}} = \frac{1}{L^3} \int_V \rho(\underline{x}) \exp(-i\underline{k} \cdot \underline{x})\, d\underline{x}, \quad \text{normalization } \frac{1}{L^3} \tag{504}$$

General properties of the periodic Fourier series:

$$\begin{aligned} \text{orthogonality } \frac{1}{L^3} \int_V \exp\left(-i\left(\underline{k} - \underline{k}'\right) \cdot \underline{x}\right) d\underline{x} &= \delta_{\underline{k},\underline{k}'}, \\ \text{closure } \frac{1}{L^3} \sum_{\underline{k}} \exp(i\underline{k} \cdot \underline{x}) &= \delta(\underline{x}) \end{aligned} \tag{505}$$

11.1.0.4 Solution to the Poisson equation in Fourier space

Replacing the density and potential by their corresponding Fourier series in the Poisson equation - assuming periodic boundaries - yields

$$\begin{aligned} \underline{\nabla}^2 \sum_{\underline{k}} \phi_{\underline{k}} \exp(i\underline{k} \cdot \underline{x}) &= 4\pi G \sum_{\underline{k}} \hat{\rho}_{\underline{k}} \exp(i\underline{k} \cdot \underline{x}) \\ \rightarrow \phi_{\underline{k}} &= -\frac{4\pi G}{\underline{k}^2} \hat{\rho}_{\underline{k}} = g_{\underline{k}} \cdot \hat{\rho}_{\underline{k}} \end{aligned} \tag{506}$$

where $g_{\underline{k}}$ is the Greens function of the Poisson equation in periodic fourier space.

11.2 Discrete Fourier Transform (DFT)

Periodicity in space led to discrete wave vectors $\underline{k}$, discretization in space will limit the different $\underline{k}$ necessary to fully represent a density distribution, leading to a discrete Fourier and inverse Fourier transform by summation.

On a computer, the density and potential in normal space are also discretized to the following positions

$$\underline{x}_{\underline{p}} = \frac{L}{N}\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} = \frac{L}{N}\underline{p}, \qquad \text{box length } L, \qquad \text{points per dimension } N, \tag{507}$$
$$p_1, p_2, p_3 \in \{0, 1, \ldots, N-1\}, \quad \rho_{\underline{p}} = \rho\left(\underline{x}_{\underline{p}}\right)$$

Using $d\underline{x} \to \left(\frac{L}{N}\right)^3$ the integral for calculating the Fourier transform of the density (eq. 504) turns into the sum

$$\hat{\rho}_{\underline{k}} = \frac{1}{N^3}\sum_{\underline{p}} \rho_{\underline{p}} \exp\left(-i\underline{k}\cdot\underline{x}_{\underline{p}}\right) \tag{508}$$

11.2.0.1 The spatial discretization (and periodicity) limits the number of $\underline{k}$ vectors that lead to possibly different $\hat{\rho}_{\underline{k}}$

$$\underline{k} \in \frac{2\pi}{L}\begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} \xrightarrow[\text{discretization of space}]{} \underline{k} = \frac{2\pi}{L}\begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = \frac{2\pi}{L}\underline{l}, \quad l_1, l_2, l_3 \in \{0, 1, \ldots, N-1\} \tag{509}$$

Why does a longer range of l_1, l_2, l_3 not introduce additional modes?: Consider the inverse Fourier transform

$$\rho(\underline{x}) = \sum_{\underline{l}} \hat{\rho}_{\underline{k}} \exp\left(i\underline{k}_{\underline{l}} \cdot x_{\underline{p}}\right) \tag{510}$$

with $\underline{x}_{\underline{p}} = \frac{L}{N}\underline{p}$. Consider now we would introduce a further $\underline{k}$ to the sum with e.g.

$$\underline{k} = \frac{2\pi}{L} \begin{pmatrix} l_1 + N \\ l_2 \\ l_3 \end{pmatrix} \tag{511}$$

then it will indeed not add a **new** mode to the sum 510, as

$$\exp\left(i\left(\frac{2\pi}{L}l_1 + \frac{2\pi N}{L}\right)\frac{L}{N}p_1\right) = \exp\left(i\frac{2\pi}{L}l_1\frac{L}{N}p_1\right)\underbrace{\exp\left(i\frac{2\pi N}{L}\frac{L}{N}p_1\right)}_{=1} \tag{512}$$

A finite number of wave vectors is sufficient to discribe a density distribution on a discretized grid - higher *frequencies* are aliased to lower ones.

11.2.0.2 Resulting discrete Fourier transform

$$\boxed{\begin{aligned} \text{Fourier space } \hat{\rho}_{\underline{l}} &= \frac{1}{N^3}\sum_{\underline{p}} \rho_{\underline{p}} \exp\left(-i\frac{2\pi}{N}\underline{l} \cdot \underline{p}\right) \\ \text{Real space } \rho_{\underline{p}} &= \sum_{\underline{l}} \hat{\rho}_{\underline{l}} \exp\left(i\frac{2\pi}{N}\underline{l} \cdot \underline{p}\right) \\ \underline{l}, \underline{p} &\in \begin{pmatrix} \{0, 1, \ldots, N-1\} \\ \{0, 1, \ldots, N-1\} \\ \{0, 1, \ldots, N-1\} \end{pmatrix} \end{aligned}} \tag{513}$$

where (in 3D) we invertibly and linearly map N^3 values $\rho_{\underline{p}}$ to N^3 values $\hat{\rho}_{\underline{l}}$.

11.2.0.3 Different conventions for the wave vector $\underline{k}$ and Nyquist frequency

The wave vectors $\underline{k}$ (in the general context of DFT also called *frequencies*) are conventionally defined as

$$\underline{k} = \frac{2\pi}{L}\begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = \frac{2\pi}{L}\underline{l}, \quad l_1, l_2, l_3 \in \left\{-\frac{N}{2}, \ldots, -1, 0, 1, \ldots, \frac{N}{2} - 1\right\} \tag{514}$$

Note: This does not make a difference compared to $l_1, l_2, l_3 \in 0, 1, \dots, N-1$ as shifts by $\frac{2\pi N}{L}$ in an entry of $\underline{k}$ do not lead to new unique modes.

This shift makes the occurence of positive and negative wave vectors explicit and the $\underline{k}$-vectors are now arranged quasi-symmetrically around $\underline{k} = (0, 0, 0)^T$, as illustrated in figure 89.

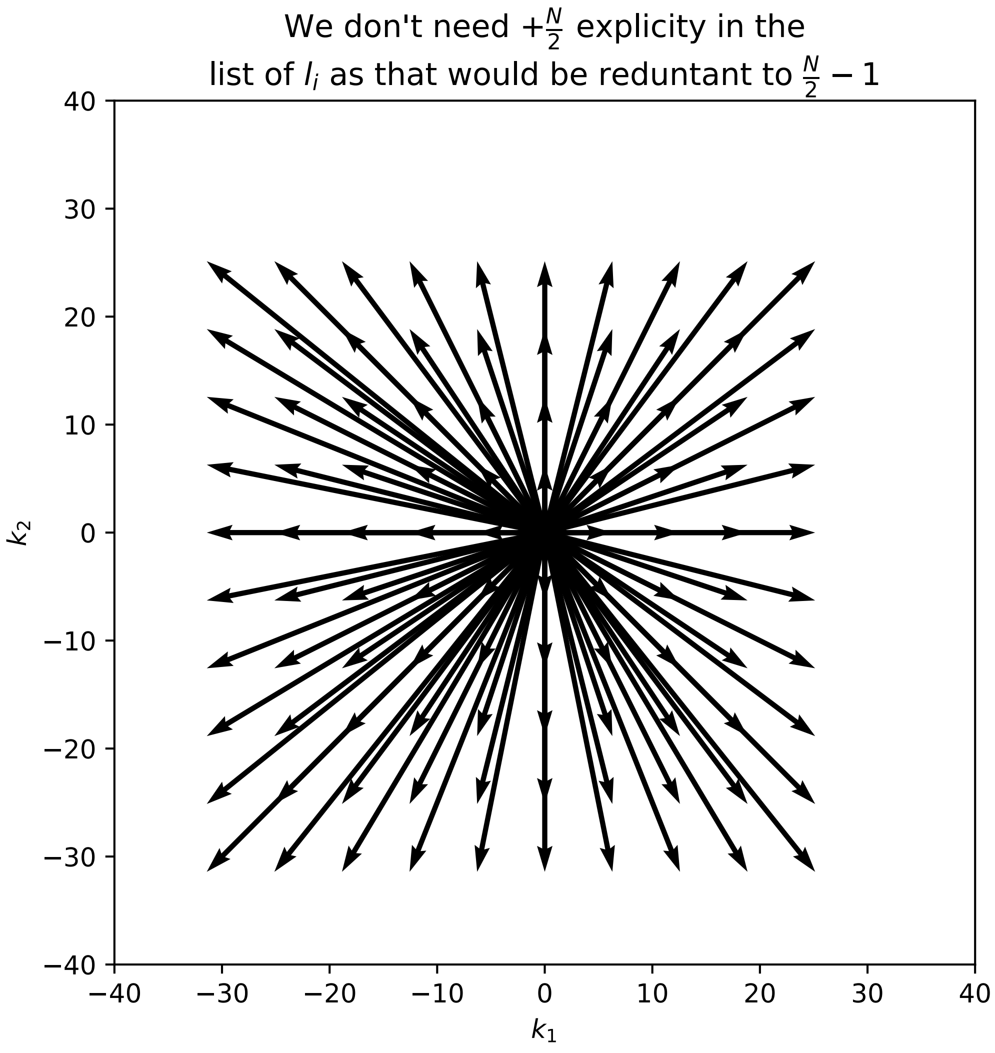


Figure 89: The wave vectors $\underline{k}$ are conventionally defined as $\underline{k} = \frac{2\pi}{L}\begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = \frac{2\pi}{L}\underline{l}$ with $l_1, l_2, l_3 \in \left\{-\frac{N}{2}, \dots, -1, 0, 1, \dots, \frac{N}{2} - 1\right\}$.

Higher frequencies than the

$$\text{Nyquist frequency } \underline{k}_{\max} = \frac{N}{2}\frac{2\pi}{L} \tag{515}$$

cannot be represented unambiguously on a grid with N points per dimension, instead they are aliased to lower frequencies, as illustrated in figure 90.

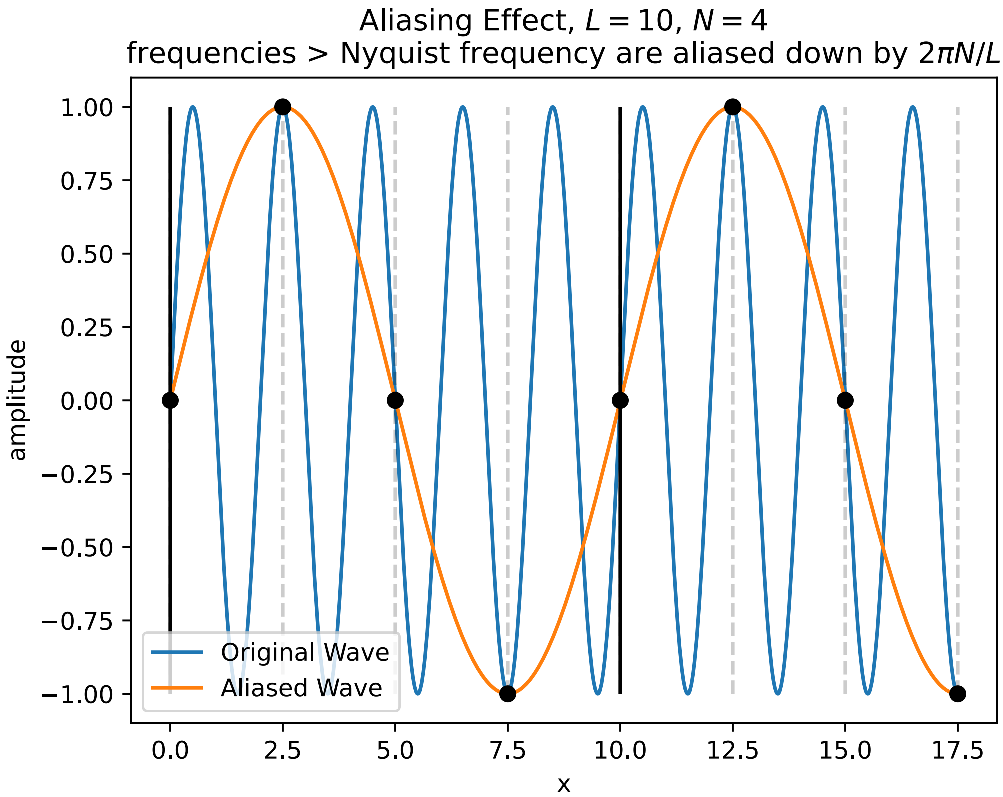


Figure 90: Higher frequencies than the Nyquist frequency $\underline{k}_{\max} = \frac{N}{2}\frac{2\pi}{L}$ cannot be represented unambiguously on a grid with N points per dimension, instead they are aliased to lower frequencies.

11.2.0.4 Plancherel's theorem

Plancherel's theorem makes the following relation

$$\sum_{\underline{p}} \left|\rho_{\underline{p}}\right|^2 = N^3 \sum_{\underline{l}} \left|\hat{\rho}_{\underline{l}}\right|^2 \tag{516}$$

- a function and its Fourier transform have the same L2 norm. We can therefore decide if we want to calculate this L2 norm in real or Fourier space.

11.2.0.5 Normalization factor

The $\frac{1}{N^3}$ normalization is set arbitrarily (could also be split between both equations), often omitted in numerical packages.

Synthesis	**Analysis**
Given $\hat{z}_0, \dots, \hat{z}_{N-1} \in \mathbb{Z}$, we search for the $$z_i = \sum_{j=0}^{N-1} \hat{z}_j \omega_N^{ij} \quad \forall i \quad \in [0 : N-1] \quad (517)$$	Given $z_0, \dots, z_{N-1} \in \mathbb{Z}$, we search for the $$\hat{z}_j = \sum_{j=0}^{N-1} z_i \omega_N^{*ij} \quad \forall \quad i \in [0 : N-1] \quad (518)$$

Table 16: Synthesis and analysis equations for the Fourier transform. ω_N are the square roots of unity (as $\omega_N^N = 1$), $\omega_N = \exp\left(\frac{2\pi \beth}{N}\right)$, where $\beth$ is the imaginary unit (to not confuse it with the index i). $\omega_N^{ij} = \exp\left(\frac{2\pi\beth}{N} ij\right)$. This can also be written as the matrix equation $\underline{z} = \underline{\underline{F}}\,\underline{\hat{z}}$ with the Fourier matrix $\underline{\underline{F}}$, where $F_{ij} = \omega_N^{ij}$.

Commonly the form of table 16 is used.

Note: In this case $\mathcal{F}^{-1}(\mathcal{F}(\underline{z})) = N^3 \underline{z}$

Higher dimensional transforms are just the cartesian products of one-dimensional transforms.

11.2.1 Computational Complexity and Fast Fourier Transform (FFT)

In the naive approach, to calculate all $z_0, \dots, z_{N-1}$ (with each calculation being $\mathcal{O}(N)$) we would need $\mathcal{O}(N^2)$ operations.

FFT-idea in a nutshell: The central ideas are to *is divide and conquer* and smartly use the unit square root. Let N be a power of 2 and $N = 2m$. Then $\omega_N^2 = \omega_m$. We can then split the sum

$$z_i = \sum_{j=0}^{N-1} \hat{z}_j \omega_n^{ij} = \underbrace{\sum_{k=0}^{m-1} \hat{z}_{2k} \omega_m^{ik}}_{:= x_i} + \omega_N^i \underbrace{\sum_{k=0}^{m-1} \hat{z}_{2k+1} \omega_m^{ik}}_{:= y_i}, \qquad \text{using } \omega_N^{i+1} = \omega_n \omega_N^i \tag{519}$$

and find

$$z_i = x_i + \omega_N^i y_i, \qquad z_{i+m} = x_i - \omega_N^i y_i \tag{520}$$

so by recursion we can construct the z_i in $\mathcal{O}(N \log N)$ operations.

11.3 DFT storage conventions

Let us get back to the frame of the DFT in the form

$$\boxed{\begin{array}{c} \text{Fourier space } \hat{\rho}_{\underline{l}} = \dfrac{1}{N^3} \displaystyle\sum_{\underline{p}} \rho_{\underline{p}} \exp\left(-i\frac{2\pi}{N}\underline{l}\cdot\underline{p}\right) \\ \text{Real space } \rho_{\underline{p}} = \displaystyle\sum_{\underline{l}} \hat{\rho}_{\underline{l}} \exp\left(i\frac{2\pi}{N}\underline{l}\cdot\underline{p}\right) \\ \underline{l} \in \left(\begin{array}{c} \left\{-\frac{N}{2},\ldots,-1,0,1,\ldots,\frac{N}{2}-1\right\} \\ \left\{-\frac{N}{2},\ldots,-1,0,1,\ldots,\frac{N}{2}-1\right\} \\ \left\{-\frac{N}{2},\ldots,-1,0,1,\ldots,\frac{N}{2}-1\right\} \end{array}\right), \quad \underline{p} \in \left(\begin{array}{c} \{0,1,\ldots,N-1\} \\ \{0,1,\ldots,N-1\} \\ \{0,1,\ldots,N-1\} \end{array}\right) \end{array}} \tag{521}$$

Both the original data and the Fourier transform are stored as simple arrays (1D Fourier). Negative *frequencies* are stored *backwards.* A 1D illustration is given in figure 91. A 2D illustration is given in figure 92. One can calculate

$$\#\text{ independent numbers in 2D grid } = 2\left(\frac{N}{2}-1\right)^2 \cdot 2 + 4\left(\frac{N}{2}-1\right)\cdot 2 + 4 = N^2 \tag{522}$$

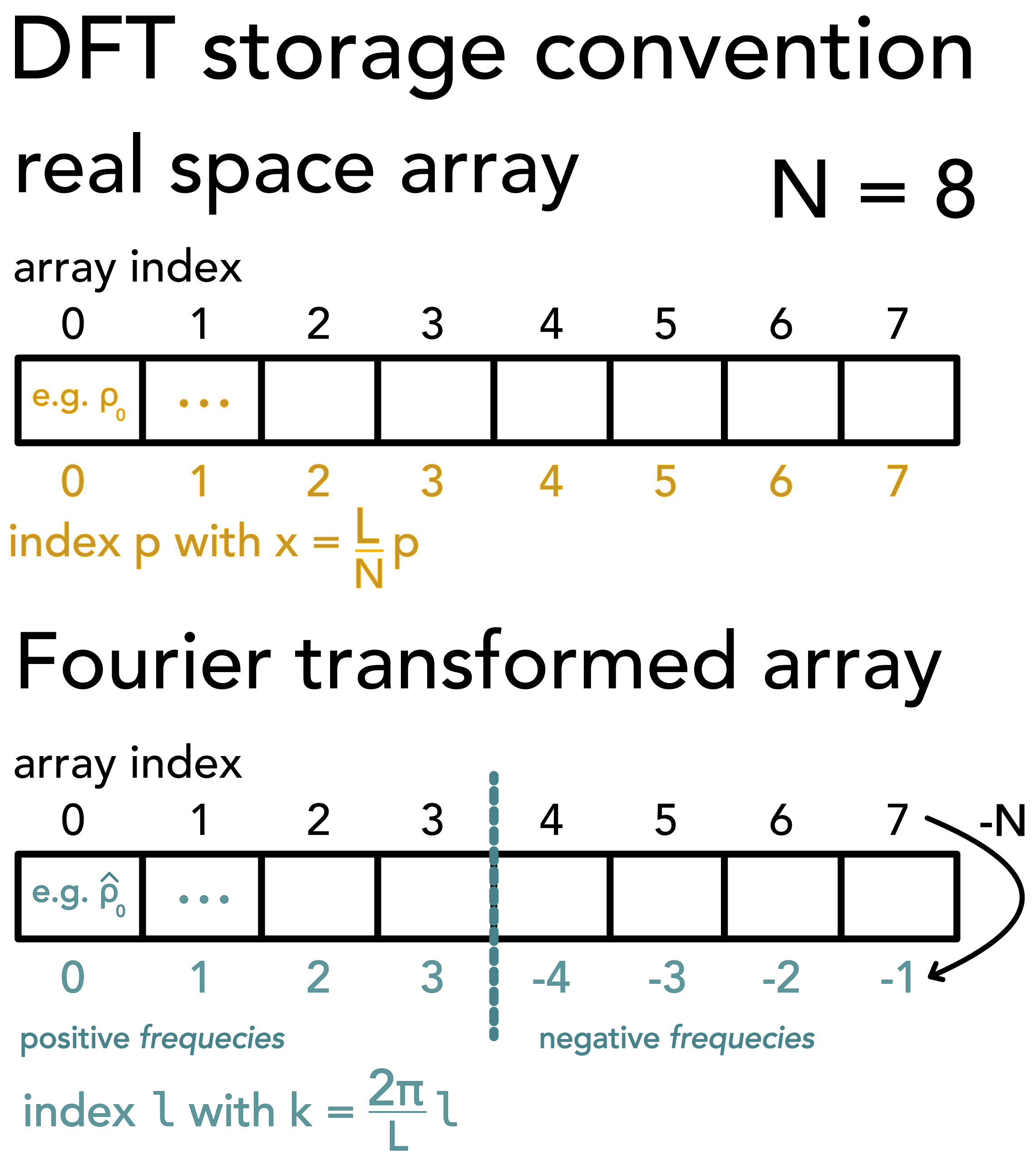


Figure 91: Illustration of the storage of the original data and the Fourier transform in 1D.

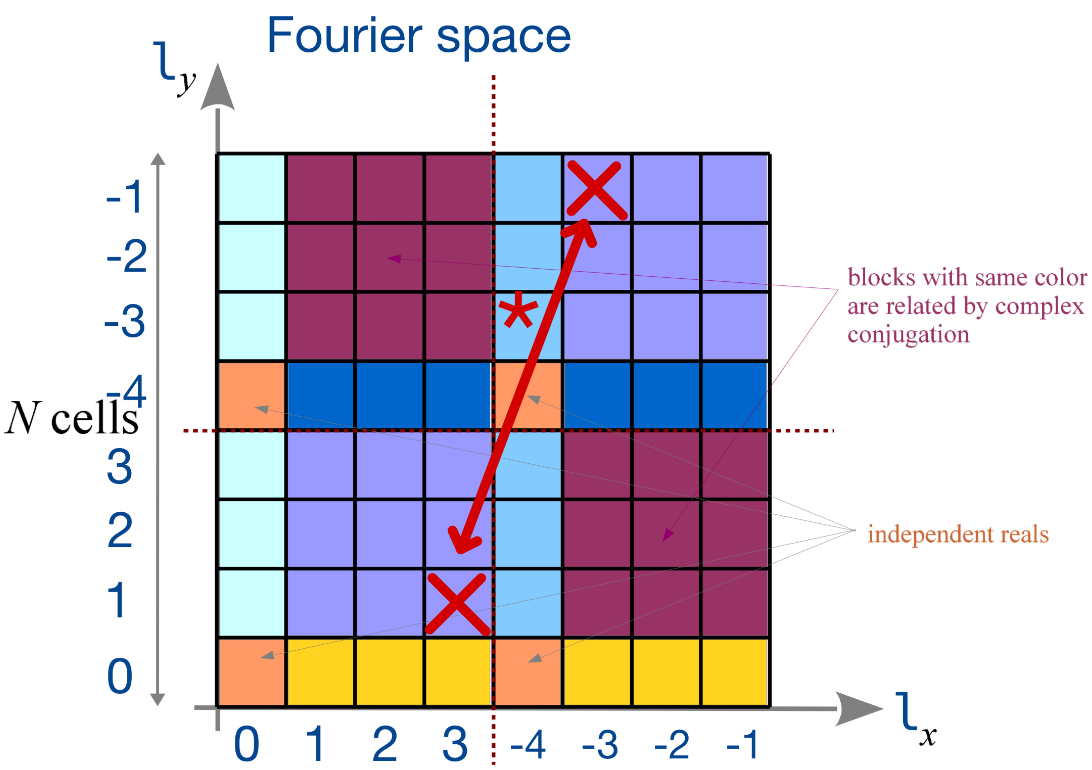


Figure 92: 2D fourier space storage convention. Note that if the data in real space is real $\rho_{\underline{p}} \in \mathbb{R}$, then $\hat{\rho}_{\underline{l}} = \hat{\rho}^*_{-\underline{l}}$. Because of aliasing $\hat{\rho}_{-\frac{N}{2}} = \hat{\rho}_{\frac{N}{2}} = \hat{\rho}^*_{-\frac{N}{2}}$ so $\in \mathbb{R}$, marked in orange in the illustration. We have (a complex number consists of two independent real numbers) the same number of independent numbers as in real space.

11.4 DFT and Linear and Cyclic Convolution

Let us step back to the convolution of two functions - e.g. for solving the Poisson equation or for smoothing an image. The convolution is defined as

$$(f \star g)(t) = \int_{\mathbb{R}} f(\tau)g(t-\tau)d\tau \tag{523}$$

so naturally for discrete f, g, we define

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n-m] \tag{524}$$

The basic intuition for convolutions is that one function is flipped and slides over the other as a window, as illustrated in figure 93.

Note: When one convolves using $\text{ifft}(\text{fft}(f) \cdot \text{fft}(g))$ this is actually the cyclic convolution

$$(f * g_N)[n] = \sum_{m=0}^{N-1} f[m]g_N[n-m] \tag{525}$$

assuming f, g are periodic with period N. The wanted result from a linear convolution is illustrated in figure 94, what we get from the cyclic convolution is illustrated in figure 95. Code for the cyclic convolution is given in code 3.

Idea: If we zero-pad sufficiently, the cyclic convolution becomes a linear convolution. For two vectors $\underline{f}$ of length N and $\underline{g}$ of length M we append zeros to both so they are both of length $N + M - 1$ (the length of the linear convolution), see figure 96 and code 4.

```
# cyclic convolution in real space
function DirectCyclicConv1D(f,g)
    N = length(f)
    Conv = zeros(N)

    for n in 1:N, m in 1:N
        if n-m+1 > 0
            Conv[n] = Conv[n] + f[m] * g[n-m+1]
        else
            # make g periodic
            Conv[n] = Conv[n] + f[m] * g[N+(n-m+1)]
        end
    end

    return Conv

end

# cyclic convolution in Fourier space
FastCyclicConv1D(f,g) = ifft(fft(f).*fft(g))
```

Code-Snippet 3: Cyclic convolution in Julia.

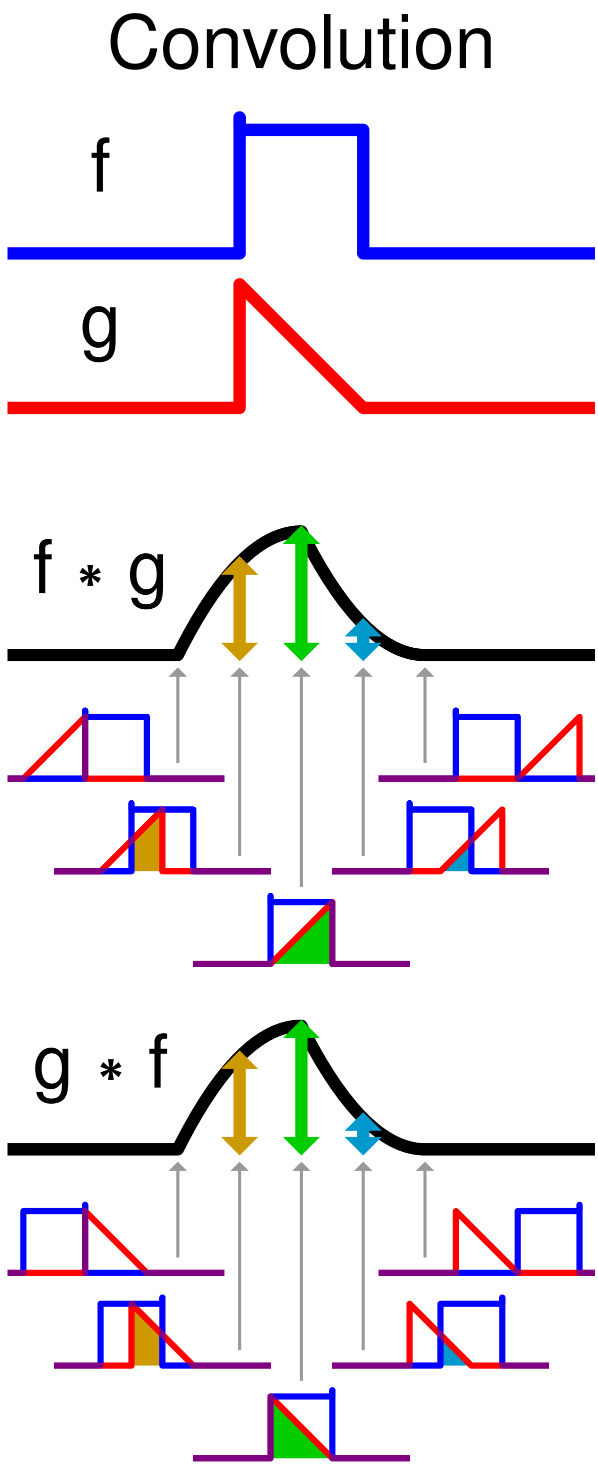


Figure 93: Basic intuition for convoltions.

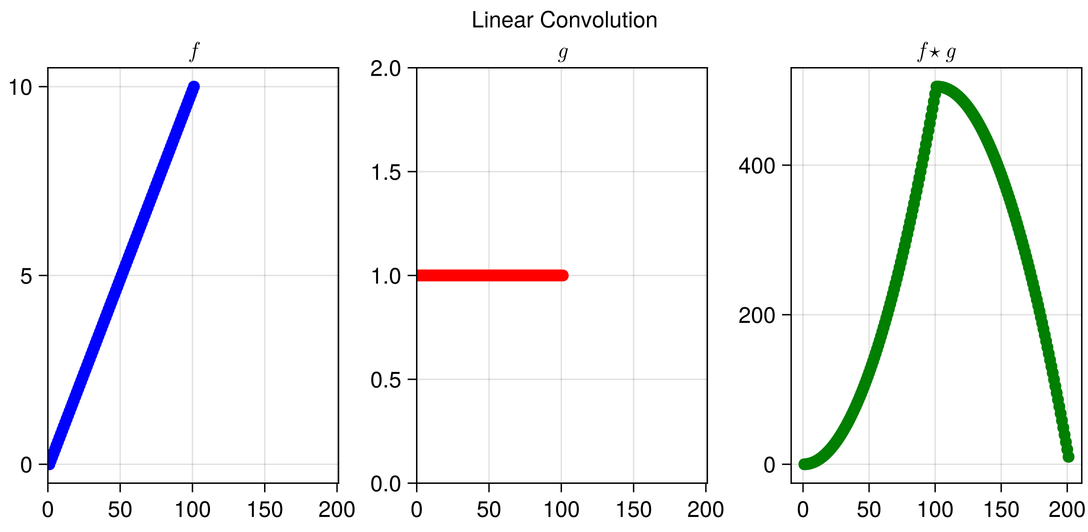


Figure 94: Example of a linear convolution.

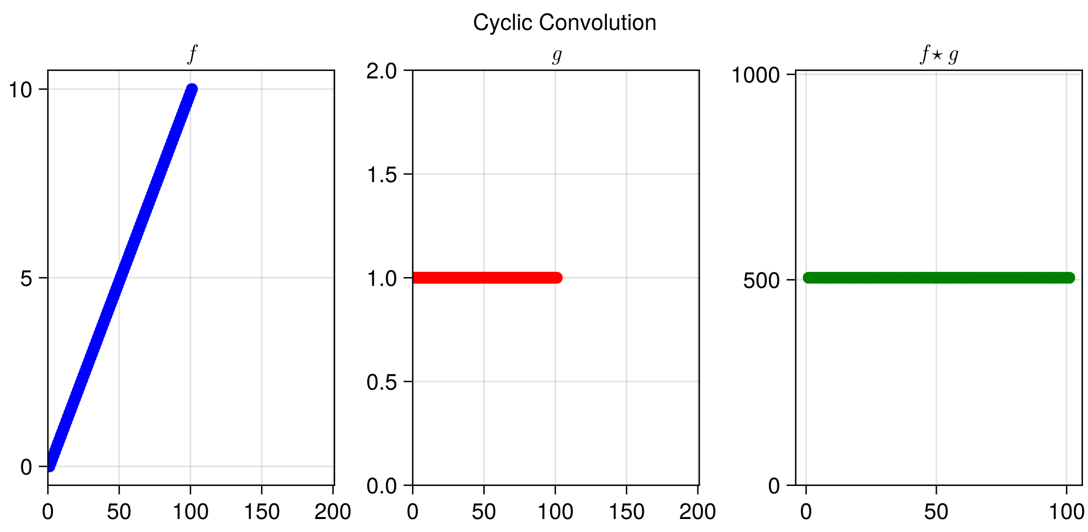


Figure 95: Example of a cyclic convolution.

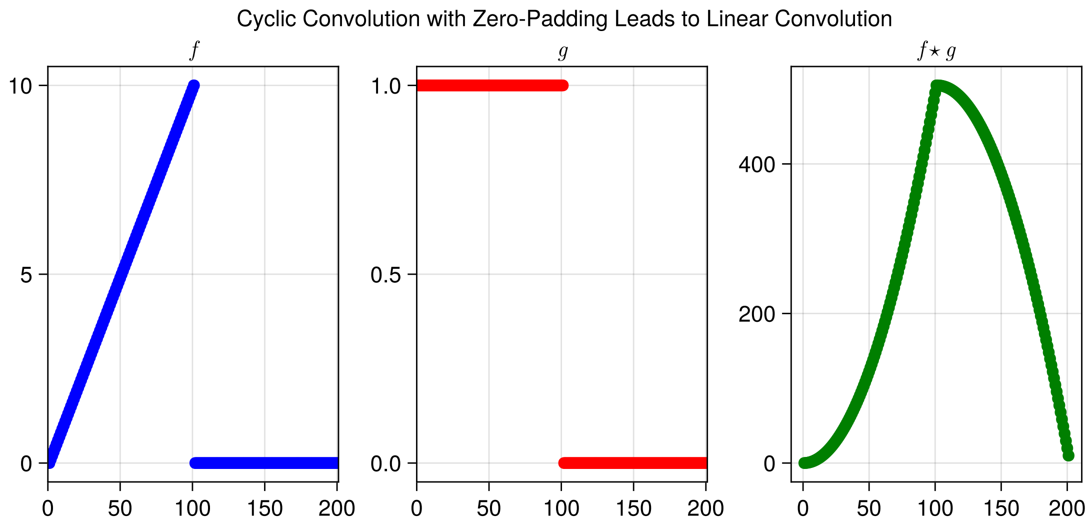


Figure 96: Illustration of the cyclic convolution becoming a linear convolution by zero-padding.

```
# simple linear convolution in real space
function conv1d(f, g)
    n = length(f)
    m = length(g)
    y = zeros(n + m - 1)
    for i in 1:n
        for j in 1:m
            y[i + j - 1] += f[i] * g[j]
        end
    end
    return y
end

# zero padded linear convolution in real space
# note that this zero padded version is equivalent
# to the cyclic convolution with zero padding
function DirectLinearConvolution(f,g)
    N = length(f)
    M = length(g)

    g_pad = [ g; zeros(N-1) ]      # length N+M-1
    f_pad = [ f; zeros(M-1) ]      # length N+M-1

    Conv = zeros(N+M-1)
    for n=1:N+M-1
        for m=1:N+M-1
            if n-m+1 > 0
                Conv[n] = Conv[n] + f_pad[m] * g_pad[n-m+1]
            end
            # n+1 <= m
        end
    end
    return Conv
end

# fast linear convolution in Fourier space
function FastLinearConvolution(f,g)
    N = length(f)
    M = length(g)
    f_pad = [ f; zeros(M-1) ]
    g_pad = [ g; zeros(N-1) ]
    return FastCyclicConv1D( f_pad, g_pad )
end
```

Code-Snippet 4: Linear convolution in Julia.

11.5 Non-periodic problems with *zero-padding* in 2D

Problem: DFT / FFT are defined for periodic problems, a convolution based on them is a cyclic convolution.

Idea: As before we can zero-pad. The cost, however, rises ($\times 4$ in 2D).

For solving the Poisson equation for non-periodic density distributions, we

1. set up the mesh so that the density distribution only lives in one quarter - rest zeroes (see figure 97).
2. periodically extend the Green's function over the whole grid (so that the distance used is the one to the nearest periodic image)

$$g_{N-i,j} = g_{i,N-j} = g_{N-i,N-j} = g_{i,j}, \quad 0 \leq i, j \leq \frac{N}{2} \tag{526}$$

   which is symmetric around the origin (when replicating the mesh in all directions). Why does the Greens function have to be extended?

3. calculate $\phi = g \star \rho$, in real space this would be

$$\phi_{\underline{p}} = \sum_{\underline{n}} g_{\underline{p}-\underline{n}} \rho_{\underline{n}}, \quad \underline{g}, \underline{\rho} \text{ periodic in the index } N \tag{527}$$

   The zero padding is large enough so that if *the Green's function slides over the density, there is no cross talk between periodic images of the density distribution.* We use the fast cyclic convolution in Fourier space.

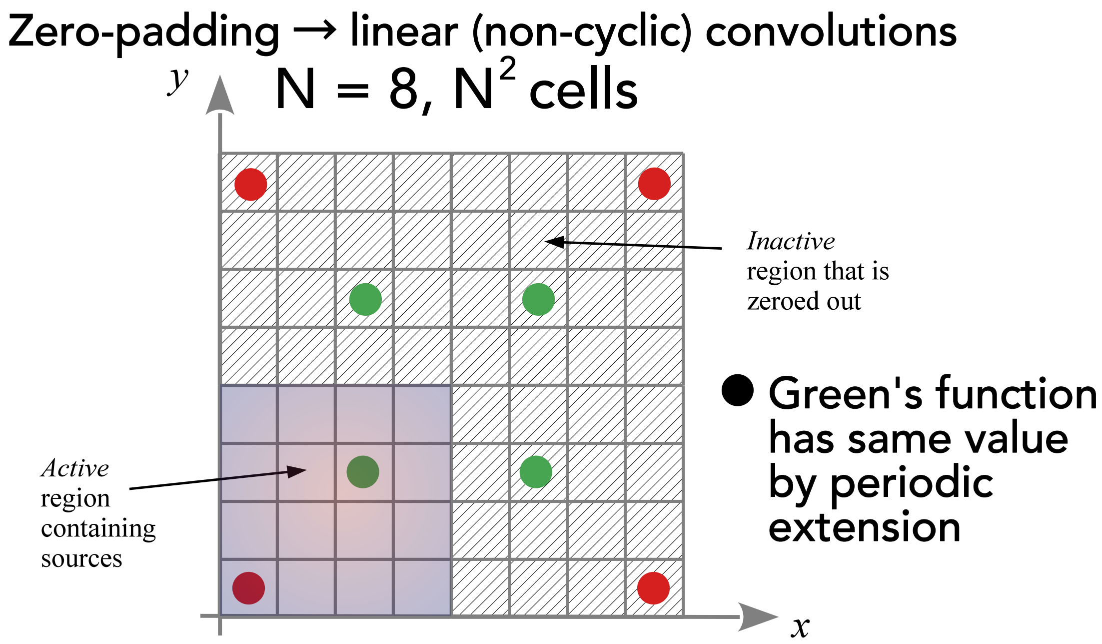


Figure 97: Illustration of the zero padding in 2D.

11.6 Power spectra and correlation functions

Power spectra and correlation functions are deeply connected via the Fourier transform. Assume the field ρ to be isotropic and homogeneous (mean or variance do not change under rotations and tranlations).

11.6.1 Definition of the power spectrum

Consider the fourier transform (decomposing the scaler field ρ into plane waves)

$$\rho(\underline{x}) = \frac{1}{(2\pi)^3} \int \hat{\rho}(\underline{k}) e^{-i\underline{k}\cdot\underline{x}} d^3\underline{k}, \quad \hat{\rho}(\underline{k}) = \int \rho(\underline{x}) e^{i\underline{k}\cdot\underline{x}} d^3\underline{x} \tag{528}$$

the variance in Fourier space defines the power spectrum $P(k)$

$$\left\langle \hat{\rho}(\underline{k})\hat{\rho}^*\left(\underline{k}'\right)\right\rangle \equiv (2\pi)^3 P(k)\delta\left(\underline{k}-\underline{k}'\right), \quad k = |\underline{k}|, \quad \text{(isotropic) power spectrum } P \tag{529}$$

- modes with different wave vector are uncorrelated (ensured by Delta distribution) to ensure homogeneity. To ensure isotropy, the power spectrum P cannot depend on the direction of $\underline{k}$.

Usually the energy spectral density for a measurement $x(t)$ over time is just $\bar{S}_{xx}(f) \triangleq |\hat{x}(f)|^2$ where f is a frequency (Fourier transform in time) and the general signal processing understanding of energy as $E \triangleq \int_{-\infty}^{\infty} |x(t)|^2 dt$ is used (which based on Plancherel's theorem $\int_{-\infty}^{\infty} |x(t)|^2 dt = \int_{-\infty}^{\infty} |\hat{x}(f)|^2 df$ makes the formula for the *spectral* energy density (in a small frequency interval) plausible).

11.6.2 Definition of the correlation function

The auto-correlation function

$$\begin{gathered}\xi(y) = \langle \rho(\underline{x})\rho(\underline{x}+\underline{y})\rangle \\ \text{average over all positions } \underline{x} and \textbf{orientations} \text{ of } \underline{y}\end{gathered} \tag{530}$$

measures the coherence of the fluctuating field ρ at all points seperated by $\underline{y}$ (can also not depend on the direction of $\underline{y}$ as of isotropy). For $\underline{y} = 0$ we have the variance.

11.6.3 Connection by Fourier Transform

Correlation function and power spectrum are Fourier transforms of each other (can be seen by inserting the Fourier tranforms into the correlation function), hence carry an equivalent amount of information.

From this we can find the variance of ρ in real space (at $y = 0$) to be

$$\sigma^2 = \frac{4\pi}{(2\pi)^3} \int P(k)k^2\, dk \tag{531}$$

11.6.4 Power function and variance of a smoothed field

Consider the smoothed field

$$\bar{\rho}(\underline{x}) = \int \rho(\underline{y}) W_R(||\underline{x} - \underline{y}||)\, d^3\underline{y}, \quad \text{window with compact support } R \tag{532}$$

which is a convolution. As of the convolution theorem $\widehat{f \star g} = \hat{f}\hat{g}$, we have

$$\hat{\bar{\rho}}(\underline{k}) = \hat{\rho}(\underline{k})\hat{W}_R(\underline{k}) \tag{533}$$

so the power spectrum (as of eq. 529) of the smoothed field is

$$\bar{P}(k) = P(k)\hat{W}_R^2(k) \tag{534}$$

which can be used to calculate a variance of the smoothed field using eq. 531.

11.7 Projections in Fourier space

Splitting a vector field into a source-free rotational and a curl-free part can be useful (e.g. for cleaning divergence from a numerically obtained magnetic field) - this can be done and proven in Fourier space.

One can proof the Helmholtz decomposition (fundamental theorem of vector analysis) using Fourier transform.

11.7.1 Fundamental theorem of vector analysis | Helmholtz decomposition

Let $\underline{F}(\underline{x})$ be a vector field with compact support and derivatives that vanish at infinity.

Then $\underline{F}$ can be uniquely decomposed into a curl-free (longitudinal, purely radial in Fourier space) and a divergence-free part

$$\begin{aligned} \underline{F}(\underline{x}) &= \underline{F}_{\perp}(\underline{x}) + \underline{F}_{\parallel}(\underline{x}) \\ \underline{\nabla} \cdot \underline{F}_{\perp} = 0, &\quad \underline{\nabla} \times \underline{F}_{\parallel} = 0 \end{aligned} \tag{535}$$

11.7.2 Proof in Fourier space

Consider a decomposition of the Fourier transform $\underline{\hat{F}}(\underline{k})$ of $\underline{F}(\underline{x})$ into a part parallel and perpendicular to $\underline{k}$ (the wave vector)

$$\begin{aligned} \underline{\hat{F}}(\underline{k}) &= \underline{\hat{F}}_{\perp}(\underline{k}) + \underline{\hat{F}}_{\parallel}(\underline{k}) \\ \underline{\hat{F}}_{\parallel}(\underline{k}) &= \underline{\hat{e}}_k \left(\underline{\hat{e}}_k \cdot \underline{\hat{F}}(\underline{k}) \right), \quad \underline{\hat{e}}_k = \frac{\underline{k}}{||\underline{k}||} \\ \underline{\hat{F}}_{\perp}(\underline{k}) &= \underline{\hat{F}}(\underline{k}) - \underline{\hat{F}}_{\parallel}(\underline{k}) \end{aligned} \tag{536}$$

In Fourier space divergence and curl are simple to calculate (they become scalar products and cross products with $\underline{k}$ respectively).

$$\underline{k} \cdot \underline{\hat{F}}_{\perp}(\underline{k}) = 0, \quad \underline{k} \times \underline{\hat{F}}_{\parallel}(\underline{k}) = 0 \tag{537}$$

from which by inverse Fourier transform the Helmholtz decomposition follows.

11.7.3 Applications of the Helmholtz decomposition

- Retain $\underline{\nabla} \cdot \underline{B} = 0$ in simulations by projecting out the radial component (the one with divergence in real space) in Fourier space ($\hat{\underline{B}}(\underline{k}) \to \hat{\underline{B}}(\underline{k}) - \underline{\hat{e}}_k \underline{\hat{e}}_k \cdot \hat{\underline{B}}(\underline{k})$, then apply inverser Fourier transform to get the source-clensed field).
- Analyze turbulent flow for shocks
- Stability analysis in astrophysics: Based on determining how much compressive motion and how much solenoidal motion is present, stability can be analyzed. Solenoidal motion stabilizes, compressive motion increases density so astrophysically the cooling rate of the system ($\propto \rho^2$) and finally leads to collapse. This can be studied by Fourier transform (problem: spatial information is non-local in Fourier space), and also based on
$$\partial_t \rho = -\rho \underline{\nabla} \cdot \underline{v}, \quad \text{solenoidal } \underline{\nabla} \cdot \underline{v} = 0 \to \partial_t \rho = 0 \tag{538}$$
which can be seen as a peak in the probability distribution of the density which is broader for compressive motion.

11.7.4 Sidenote: Importance of dimensionality

Depending on the dimensionality, we have

- 1D: one compressible mode
- 2D: one compressible, one source-free (solenoidal, non-compressive) mode
- 3D: one compressible, two source-free modes

In a tube with small width we mostly have compressible modes, in shallow water we can have vortices with spin perpendicular to the surface.

12 Collisionless particle systems

12.1 Introduction of collisionless systems in the context of fluid modeling

Let us widen our view to collisionless systems (*collisionless »fluids«*) and recapitulate on fluid dynamics to avoid confusion.

Important fluid concepts are

- **Boltzmann:** starting from the view of lots of interacting particles one can derive the Boltzmann equation (and we will repeat on this) in the collisionless case $\left(\frac{df}{dt}\big|_{\text{coll}} = 0\right)$ called Vlasov equation
- **Navier-Stokes:** based on the moments of the Boltzmann equation one can derive a continuity, momentum and energy equation - Navier-Stokes equation for a collisional and Jeans equation for a *collisionless* fluid
- **Euler:** in case viscosity and conductivity can be ignored, the Navier stokes equations can be reduced to the Euler equations
- **Fluid variables:** *fluid elements* can be characterized by six hydrodynamical variables: density ρ, fluid velocity $\underline{u}$ (of a fluid element, not to be confused with the velocity of individual particles), pressure P and specific internal energy ϵ
- **Closure:** The Euler equations give 5 relations of the fluid variables - one continuum equation, three momentum equations (vector-equation), one energy equation; we need a further constitutive relation - almost always an equation of state $P = P(\rho, \epsilon)$
- **Perspectives:** The Eulerian view is fixed in space, in the Lagrangian view, the description co-moves with the fluid flow

where we have explored some solvers for the Navier-Stokes / Euler equation with respective closure

- **Eulerian methods:** Based on a mesh / accounting volumes, the evolution of the fluid variables is e.g. calculated based on intercell fluxes
- **Lagrangian methods:** In Smoothed Particle Hydrodynamics we introduce SPH-particles (macroscopic, describing the fluid) forwarded based on the Euler / Navier Stokes equation

Collisionless fluids are very different from our standard collisional fluids in that

- their behavior is not collision dominated, if one would want to use a description based on fluid variables, one would at least have to use a local anisotropic pressure described via a stress tensor - as there are no frequent collisions that would act to isotropize the local pressure
- no equation of state exists for a collisionless fluid, so one can never close the set of fluid equations, unless one makes a number of simplifying assumptions
- no fluid element can be defined for a collisionless fluid (one in which the continuum hypothesis would hold) - the macroscopic fluid perspective for deriving fluid laws cannot be used

Let us give examples for collisional and collisionless systems

- Systems that can be described as (collisional) fluid
  - stars - to good approximation the equation of state of a star is that of an ideal gas
  - giant gaseous planets
  - planet atmospheres
  - ...
- Collisionless systems
  - Galaxies (stellar component) - two stars in a galaxy - to a good approximation - will never collide
  - Dark matter (halos) - assumed to be collisionless

Note: Complex systems often have collisional and collisionless components: In galaxies the gaseous component can be described with classic hydrodynamic equations (enough collisions to isotropize the local pressure), but stellar and dark matter lack collisions, where one could use a local anisotropic pressure (extending classical hydrodynamic simulations) but N-body simulations have shown to be more stable. For instance Mitchell et al., 2012 write »For simulations that deal only with dark matter or stellar systems, the conventional N-body technique is fast, memory efficient and relatively simple to implement. However when extending simulations to include the effects of gas physics, mesh codes are at a distinct disadvantage compared to Smooth Particle Hydrodynamics (SPH) codes. Whereas implementing the N-body approach into SPH codes is fairly trivial, the particle-mesh technique used in mesh codes to couple collisionless stars and dark matter to the gas on the mesh has a series of significant scientific and technical limitations. These include spurious entropy generation resulting from discreteness effects [...]« and introduce a method to use »collisionless Boltzmann moment equations as a means to model the collisionless material as a fluid on the mesh«.

Model reduction for collisionless systems: While we cannot directly apply the solvers discussed so far to collisionless systems, central ideas remain important: For instance to represent the system based on artificial / fiducial particles with larger masses in very high-N systems.[a] For instance in the large dark-matter-only N-body *millenium* simulation (Springel, 2005), in a cube with sidelength $2 \cdot 10^9$ lightyears, $2160^3 \approx 10^{10}$ fiducial particles where used on which 10^{18} solar masses were equally distributed.

[a] A low N system would be the planets in the solar system, a medium-N system stellar clusters with $N \sim 10^2$ stars and high N systems could be globular clusters.

12.2 Structure of the following considerations

In the following we will consider

- From a phase space view, what is a collisionless system and how can it be described?
- What systems can be assumed to be collisionless?
- N-body model of collisionless systems

12.3 General N-particle ensembles | one-, two-, three, ..., particle distribution | BBKGY chain

The exact particle distribution of N particles is given by (exact Klimontovich-Dupree representation)

$$F(\underline{x}, \underline{v}, t) = \sum_{i=1}^{N} \delta(\underline{x} - \underline{x}_i(t)) \cdot \delta(\underline{v} - \underline{v}_i(t)) \tag{539}$$

Let p be the N particle phase space probability at time t

$$p\left(\underline{x}_1, \ldots, \underline{x}_N, \underline{v}_1, \ldots, \underline{v}_N\right) d\underline{x}_1 \cdots d\underline{x}_N d\underline{v}_1 \cdots d\underline{v}_N \tag{540}$$

The mean phase space density (ensemble-averaged one particle distribution) is

$$\begin{aligned} f_1(\underline{x}, \underline{v}, t) = \langle F(\underline{x}, \underline{v}, t) \rangle &= \int F(\underline{x}, \underline{v}, t) \cdot p d\underline{x}_1 \cdots d\underline{x}_N d\underline{v}_1 \cdots d\underline{v}_N \\ \underset{\text{plug in } F \to N \text{ terms}}{=} \; & N \int p\left(\underline{x}, \underline{x}_2, \ldots, \underline{x}_N, \underline{v}, \underline{v}_2, \ldots, \underline{v}_N\right) d\underline{x}_2 \cdots d\underline{x}_N d\underline{v}_2 \cdots d\underline{v}_N \end{aligned} \tag{541}$$

with $f_1(\underline{x}, \underline{v}, t)$ being the mean number of particles in the phase space volume $d\underline{x}d\underline{v}$ around $(\underline{x}, \underline{v})$. The form is quite natural, if we want the probability of one particle being at a certain phase-space coordinate, we have to integrate out the others (marginalize) (as we know from quantum mechanics).

Ensemble-averaged two-particle distribution: What do we expect the mean of the product of the number of particles at two phase-space coordinates $(\underline{x}, \underline{v})$ and $(\underline{x}', \underline{v}')$ to be?

$$\begin{aligned} f_2\left(\underline{x}, \underline{v}, \underline{x}', \underline{v}', t\right) &= \left\langle F(\underline{x}, \underline{v}, t) F\left(\underline{x}', \underline{v}', t\right)\right\rangle \\ &= N \cdot (N-1) \int p\left(\underline{x}, \underline{x}', \underline{x}_3, \ldots, \underline{x}_N, \underline{v}, \underline{v}', \underline{v}_3, \ldots, \underline{v}_N\right) d\underline{x}_3 \cdots d\underline{x}_N d\underline{v}_3 \cdots d\underline{v}_N \end{aligned} \tag{542}$$

$f_1, f_2, f_3, \ldots$ is the so called BBGKY chain. The Boltzmann equation is e.g. a model for f_1.

12.4 Uncorrelated (collisionless) systems | multiplication closure to the BBGKY chain

Consider the simplest closure to the BBGKY hierarchy

$$f_2\left(\underline{x}, \underline{v}, \underline{x}', \underline{v}', t\right) = f_1(\underline{x}, \underline{v}, t) f_1\left(\underline{x}', \underline{v}', t\right) \tag{543}$$

i.e. we assume the particles to be uncorrelated, i.e. a particle at $\underline{x}', \underline{v}'$ does not effect one at $\underline{x}, \underline{v}$.

Note: This is akin to $P(A, B) = P(A) \cdot P(B)$ for independent events A, B.

Still particles are effected by the global effect of all others. For instance electrons in a plasma can be assumed uncorrelated.

12.4.1 General Continuity equation for probability in phase space

Let $\underline{w} = \left(\underline{x}_1, \ldots, \underline{x}_N, \underline{v}_1, \ldots, \underline{v}_N\right)$ be the phase space state, so $p = p(\underline{w})$. As the particles themselves, probability is conserved as captured in the continuity equation

$$\begin{aligned} \partial_t p + \underline{\nabla}_{\underline{w}} \cdot (p \cdot \underline{\dot{w}}) = 0 \\ \partial_t p + p \underline{\nabla}_{\underline{w}} \underline{\dot{w}} + \underline{\dot{w}} \underline{\nabla}_{\underline{w}} p = 0 \end{aligned} \tag{544}$$

so in terms of the particle velocities and positions (apply the chain rule)

$$\frac{\partial p}{\partial t} + \sum_i \left(p \frac{\partial \underline{\dot{x}}_i}{\partial \underline{x}_i} + \frac{\partial p}{\partial \underline{x}_i} \cdot \underline{\dot{x}}_i + p \frac{\partial \underline{\dot{v}}_i}{\partial \underline{v}_i} + \frac{\partial p}{\partial \underline{v}_i} \cdot \underline{\dot{v}}_i \right) = 0 \tag{545}$$

12.4.2 Liouville equation for the general evolution of phase space probability

Based on the Hamiltonian equations[22] one gets $\partial_{\underline{x}} \underline{\dot{x}} = -\partial_{\underline{v}} \underline{\dot{v}}$ and so

$$\frac{d\rho}{dt} = \partial_t p + \sum_i \left(\underline{v}_i \cdot \frac{\partial p}{\partial \underline{x}_i} + \underline{a}_i \cdot \frac{\partial p}{\partial \underline{v}_i} \right) = 0, \quad \underline{a}_i = \underline{\dot{v}}_i = \frac{F_i}{m_i} \text{ (Liouville's eqn)} \tag{546}$$

12.4.3 Vlasov equation for collisionless systems

In the collisionless / uncorrelated limit, one obtains the Vlasov equation for $f := f_1$ (multiply Lioville's equation by F and integrate)

$$\frac{\partial f}{\partial t} + \underline{v} \cdot \frac{\partial f}{\partial \underline{x}} + \underline{a} \cdot \frac{\partial f}{\partial \underline{v}} = 0 \tag{547}$$

- the phase space density along trajectories is constant in the collisionless case. We can also understand the Vlasov equation based on the Boltzmann equation in eq. 131, where we assume the fluctuations in accelerations and change of phase space density to be decoupled.

[22] $\underline{\dot{x}} = \partial_t \underline{x} = \partial_{\underline{p}} H, \quad \underline{\dot{p}} = -\partial_{\underline{x}} H$

12.4.4 Accelerations in collisionless systems - including gravity into the Vlasov equation

Collective effects like self gravity can still be described in collisionless systems. Based on the density

$$\rho(\underline{x}, t) = m \int f(\underline{x}, \underline{v}, t) d\underline{v} \tag{548}$$

we can calculate the gravitational potential via Poisson's equation and thus an acceleration

$$\underline{\nabla}^2 \phi = 4\pi G \rho \rightarrow \underline{a} = -\underline{\nabla}_{\underline{x}} \phi \tag{549}$$

so the Vlasov equation

$$\frac{\partial f}{\partial t} + \underline{v} \cdot \frac{\partial f}{\partial \underline{x}} - \underline{\nabla}_{\underline{x}} \phi \cdot \frac{\partial f}{\partial \underline{v}} = 0 \tag{550}$$

As of the Vlasov equation we do not describe single particles anymore but rather directly model the phase space density. For discretization we later reintroduce particles - but not the physical ones rather macro-particles sampling the phase space in a Monte-Carlo fashion (Monte-Carlo comes later).

12.5 When is a gravitational system collisionless?

Idea: In every system collisions happen, but if the time of our simulation is shorter than the timescale on which collisions have a meaningful impact t_{relax} we can assume the system to be collisionless. **We thus view a system as collisionless if**

$$\begin{aligned} t_{\text{relax}} \gg t_{\text{of interest for the simulation}}, &\quad \text{final result } t_{\text{relax}} = \frac{N}{8 \log N} t_{cross} \\ \text{number of particles in the system } N, &\quad \text{typical time to cross the system } t_{\text{cross}} \end{aligned} \tag{551}$$

Where we will derive and explain t_{relax} in the following. We can already see the surprising result: **For two gravitational systems with the same mass and size, the one with more smaller particles has a longer relaxation time so acts *more collisionless* as a whole.**

$$\text{overall potential more averaged } \gg \text{ more frequent encounters} \tag{552}$$

12.5.1 Relaxation time in a gravitational system

But what is the relaxation time t_{relax}? Consider a system of size R with N particles of (average) mass m. Consider a particle moving at speed v through it. Assume we would know the mean squared perpendicular velocity $\langle(\Delta v_\perp)^2\rangle$ a particle would accumulate as of collisions by crossing the whole system and the time t_{cross} to cross the system.

We can then reasonably define the relaxation time as

$$t_{\text{relax}} \equiv \frac{v^2}{\langle(\Delta v_\perp)^2\rangle / t_{\text{cross}}} \tag{553}$$

so the time at which perturbations have added up to a perpendicular velocity of the order of the velocity the particles move with - where collisions can surely not be neglected anymore.

Our next steps therefore are

- find an expression for t_{cross}
- find an expression for $\langle(\Delta v_\perp)^2\rangle$

12.5.2 Crossing time

Consider a system of size R with N particles. It takes the particles roughly

$$t_{\text{cross}} = \frac{R}{v} \sim \frac{R^{\frac{3}{2}}}{\sqrt{GM}} \tag{554}$$

to cross the system, where we used

$$v^2 \simeq \frac{GM}{R}, \quad M = Nm, \text{(avg.) particle mass } m \tag{555}$$

for the typical speed v of a field star is roughly that of a particle in a circular orbit at the edge of the galaxy.

Note: t_{cross} depends on the total mass and size of the system but not the individual masses directly.

12.5.3 Change in perpendicular velocity when crossing the system

For the change in perpendicular velocity, we can approximate

$$\langle (\Delta v_\perp)^2 \rangle \approx v^2 \langle \theta^2 \rangle$$
$$\text{mean squared deflection angle over the whole system } \langle \theta^2 \rangle \tag{556}$$
$$\text{typical velocity of a particle } v$$

We will

- calculate the deflection in one interactions
- calculate $\langle \theta^2 \rangle$ assuming small angle interactions
- justify that most of the total deflection stems from many small deflections, justifying using $\langle \theta^2 \rangle$ under the assumption of small angle deflections

12.5.3.1 Size of one small angle deflections based on the impact parameter

Consider the deflection scenario sketched in figure 98.

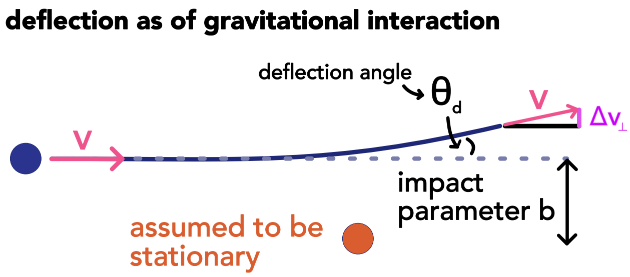


Figure 98: Gravitational deflection

In the small angle (here small deflection) approximation, we get

$$\theta_d \approx \sin \theta_d \approx \frac{\Delta v_\perp}{v} \tag{557}$$

where the change in perpendicular velocity as of the gravitational field of the other particle is calculated as if the particle would have moved in a straight line (small deflection, Born approximation)

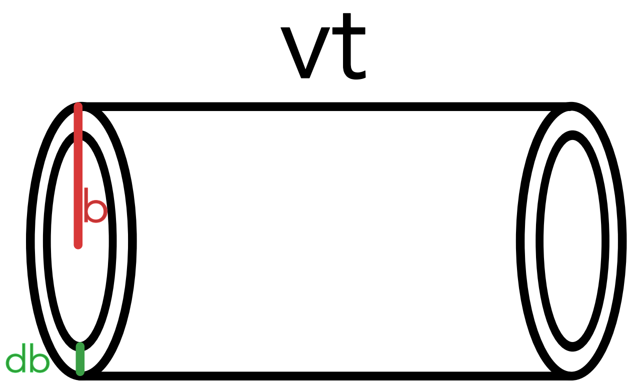


Figure 99: Impact cylinder

$$\Delta p_\perp = m\Delta v_\perp = -m \underbrace{\int_{-\infty}^{\infty} \underline{\nabla}_\perp \phi \, dt}_{\substack{\text{deflection as} \\ \text{of grav. field}}} \approx -m \int_{-\infty}^{\infty} \underbrace{\partial_b \left(\frac{Gm}{\sqrt{b^2 + v^2 t^2}} \right)}_{\substack{\text{Born} \\ \text{approximation}}} dt \\ = \cdots = \frac{2Gm^2}{bv} \tag{558}$$

so the deflection angle is

$$\theta_d \approx \frac{b_0}{b} \text{ with critical impact parameter } b_0 = \frac{2Gm}{v^2} \ll b \text{ as of our small angle approximation} \tag{559}$$

where we would have $b = b_0$ for $\Delta v_\perp = v$.

12.5.3.2 Mean squared deflection for many small deflections

Over many interaction we accumulate

$$\langle \theta^2 \rangle = \sum_{\text{all encounters}} \theta_d^2 = \sum_{\text{all encounters}} \left(\frac{b_0}{b} \right), \quad \langle \theta \rangle = 0 \text{ as of symmetry} \tag{560}$$

and write

$$\langle \theta^2 \rangle = \int_{b_{\min}}^{b_{\max}} \theta_d^2 dN, \quad \text{number of encounters } dN = n\, vt\, 2\pi b\, db \text{ with } b \in [b, b + db] \tag{561}$$

(for dN see figure 99) so (plug in and integrate)

$$\boxed{\left\langle \theta^2 \right\rangle = n 2\pi b_0^2 v t \ln\Lambda, \qquad \text{Coulomb logarithm } \ln\Lambda = \ln\frac{b_{\max}}{b_{\min}}} \tag{562}$$

12.5.3.3 Many small deflections are more important than few large ones

Note: In the following we will compare deflection frequencies. The frequency in the small angle case is the frequency with which small deflections add up to e.g. 1 or $\frac{\pi}{2}$, so that a higher frequency means that small deflection are overall more important (otherwise small deflections have a higher *frequency* without calculation).

Based on our previous result for **small deflections** we can calculate the time t until $\langle\theta^2\rangle \approx 1$ (equivalent to one deflection with $b = b_0$), to get a deflection frequency

$$\nu_d \equiv \frac{1}{t_d} = n\, 2\pi b_0^2 v \ln\Lambda \tag{563}$$

Let us continue with the **mean deflection frequency based on large deflections**. Large deflections are those for $b \lessapprox b_0$. The mean deflection frequency can be calculated as

$$\begin{gathered}\nu_{\text{d, single}} \equiv \frac{1}{t_{\text{d, single}}} = \frac{v}{\lambda_{mfp}} = n \underbrace{\sigma}_{\approx \pi b_0^2} v \approx n\pi b_0^2 v \\ \text{cross section } \sigma, \quad \text{mean free path } \lambda_{mfp}, \quad \text{number density } n\end{gathered} \tag{564}$$

so we can relate

$$\frac{\nu_d}{=}\nu_{\text{d, single}} \ln\Lambda, \quad \text{typically } 20 \leq \ln\Lambda \leq 30 \tag{565}$$

so the deflection frequency based on lots of small deflections adding up is larger than the one based on single big scattering event which based on our definition of the deflection frequency means that small angle deflections are more important. This is also reflected in

$$\frac{\sigma_{\text{many small deflections adding to } \frac{\pi}{2}}}{\sigma_{\text{one } \frac{\pi}{2} \text{ deflection}}} = 8\ln\Lambda \tag{566}$$

12.5.3.4 Finally the calculation of $\left\langle (\Delta v_\perp)^2 \right\rangle$

We can now use the result for many small deflections

$$\left\langle \theta^2 \right\rangle = n 2\pi b_0^2 v t \ln\Lambda \tag{567}$$

in

$$\langle(\Delta v_\perp)^2\rangle \approx v^2\langle\theta^2\rangle = v^2 n 2\pi b_0^2 vt \ln\Lambda \underbrace{=}_{N=vt\,\pi R^2\,n} = 2N\frac{vb_0}{R}^2 \ln\Lambda \tag{568}$$

justified by the higher meaning of many small deflections.

12.5.3.5 How to choose b_{min} and b_{max} in the Coulomb logarithm?

We choose $b_{max} = R$ (the system size) and $b_{min} \approx b_0 = \frac{2Gm}{v^2}$ (the critical impact parameter) (as large deflections with $b \lessapprox b_0$ are rare).

Using the typical velocity $v^2 \approx \frac{GM}{R}$ and $N = \frac{M}{m}$ we can write

$$\boxed{\log\lambda \approxeq \log\frac{N}{2} \approxeq \log N} \tag{569}$$

Which gets us to the initially stated result for the relaxation time.

12.5.4 Examples of astrophysical relaxation times

As a reference for determining if collisionless or not dependent on the relaxation time, we use the age of the universe $t_{age} = \frac{1}{H_0} \sim 10$ Gyr. So while here the globular star cluster is not collisionless, on a more usual timescale it is. The relaxation times are in table 17 - as previously noted systems of lots of small particles or more collisionless.

System	Number of bodies $\sim$	Crossing time $\sim$	Relaxation time $\approx \frac{N}{8\ln N}t_{\text{cross}} \sim$	Collisionless over age of universe $t_{\text{relax}} \gg t_{\text{age}}$
Globular star clusters	10^5	$0.5Myr$	$0.5Gyr$	No
Stars in typical galaxy	10^{11}	$\frac{1}{100H_0}$	$5\cdot10^6 t_{\text{age}}$	Yes
Dark matter in galaxy	10^{77}	$\frac{1}{10H_0}$	$10^{73} t_{\text{age}}$	Absolutely

Table 17: Relaxation times

12.6 N-body models of collisionless systems

Idea for modelling a collisionless system - N-body simulation: Use the standard gravitational equations of motion (not fluid equations) but for fiducial heavier macro particles than in the original system.

We introduce non-physical macro particles, to discretize the collisionless fluid described by the Poisson-Vlasov system. We use far fewer macro particles than particles in the real system - the macro particles are heavier (/ have more charge). The macro-particles follow the equations

$$\ddot{\underline{x}} = -\underline{\nabla}\phi\left(\underline{x}_i\right), \quad \phi(\underline{x}) = -G\sum_{j=1}^{N}\frac{m_j}{\left[\left(\underline{x}-\underline{x}_j\right)^2+\epsilon^2\right]^{\frac{1}{2}}} \tag{570}$$

softening length ϵ

Validity-note: The real system we model is composed of much more particles than our N and is collisionless (remember $t_{\text{relax}} = \frac{N}{8\log N}t_{cross}$). For our model to be valid, it also has to be collisionless, so with our lower N we still need to fulfill $t_{\text{relax}} \gg t_{\text{of interest for the simulation}}$ (and to have a sufficiently smooth gravitational potential (/ well described compared to the real *smooth-potential* system)).

Assume that our fiducial particles sufficiently well create the gravitational potential of the real system. Then a fiducial particle at any point, will have the same acceleration as a real particle there (the force is higher but also the inertial, cancelling to given the same acceleration) - so the fiducial particles will follow valid real-particle trajectories.

Problem: We only retrieve one (noisy) realization of the one-point function f_1 by one N-particle simulation

Idea: We can combine multiple simulation results for ensemble averages. The details of f_1 are critical. For instance, for multiple crossings of a shock front we would have a high energy tail in the velocity distribution.

12.6.1 The softening length ϵ

There is a softening length in the denominator that reduces the potential for distances very close to other particles. This is especially important if we choose large macro-particles - the softening length gives a smallest impact parameter on a macro scale.

- This avoids large angle scattering (as strong potential interaction $\rightarrow$ strong deflection)
- Avoid numerically expensive singularities: Without softening, there would be singularities in the potentials, which would cause high numerical effort when integrating the orbits (as of the large numbers)
- Avoid bound particle pairs: When gravitational particles can come very close to each other they can form highly interactive / correlated pairs - and we want a collisionless system not highly correlated particles. Bounded pairs are avoided if

$$\langle v^2 \rangle \gg \frac{Gm}{\epsilon} \tag{571}$$

> **The softening length introduces a smallest resolved scale / smallest trustworthy scale**. We must make a compromise between spatial resolution, computational cost and the points discussed above.

13 Force calculations | tree algorithms and particle mesh technique

We have discretized our N-body system using fiducial macro-particles. The equations of motion are given by

$$\ddot{\underline{x}} = -\underline{\nabla}\phi\left(\underline{x}_i\right), \quad \phi(\underline{x}) = -G\sum_{j=1}^{N}\frac{m_j}{\left[\left(\underline{x}-\underline{x}_j\right)^2+\epsilon^2\right]^{\frac{1}{2}}} \tag{572}$$

Integration in time: Given the forces on the particles, we can integrate this ordinary differential equation in time. Classically a symplectic method like leapfrog would be used. But modern N-body-packages like Rebound use high-order adaptive-step-size integrators. See section 3.7.3 for details.

But how can we calculate the forces on the particles given the particle positions?

13.1 Calculating the forces | Direct summation

We can simply and exactly calculate

$$\ddot{\underline{x}}_i = -G\sum_{j=1}^{N}\frac{m_j}{\left[\left(\underline{x}_i-\underline{x}_j\right)^2+\epsilon^2\right]^{\frac{3}{2}}}\left(\underline{x}_i-\underline{x}_j\right) \tag{573}$$

Problem: For the force on each of the N particles, $N-1$ terms have to be evaluated and summed - so in total we have $\mathcal{O}(N^2)$ operations. For t_{relax} to be large (so the simulated time can be large), N must be large. But even for $N = 10^{10}$ assuming we can to $N = 10^6$ in one month of computer time, we'd need to run our computer $\sim 10^7$ years.

13.2 Overview on faster, approximate force calculation schemes

- **tree algorithms** - distant groups do not need to be resolved in every detail to calculate their respective forces - some or even one term of the multipole expansion can be used. Generally we can construct hierarchical multipole methods where a tree is a method to partition space based on the particle distribution.
- **particle mesh methods** - from the particle distribution, a mass density on a grid is calculated and using

- **fourier transform based methods** to calculate the potential from the Poisson equation and the density distribution (the necessary convolution is a multiplication in Fourier space and FFT is fast ($\mathcal{O}(N \log N)$))
- **iterative solvers** relaxation methods to solve the Poisson equation

the forces on gridpoints can be calculated and interpolated to the particles.

We can often combine the different force calculation approaches and use direct summation on small scales. To speed things up we use GPUs that are very fast for easy parallel operations like (block) matrix multiplication.

13.3 Faster method I | Multipole expansion | tree algorithms

Idea: In tree algorithms the force on one particle is calculated based on multipole expansions of groups in the *far-field* rather than individual particles (see figure 100), while the *near-field* is calculated exactly.

Consider the simple example where we want to calculate the force (and with this the movement) of a single particle based on the interaction with a distant group of particles. If the group spans only a small angle of the particles view-field, the force acting on the particle can be approximated by the one along the axis to the groups center of mass, so

$$\underline{a}_{\text{our particle}} \approx -GM \frac{\underline{r}_{\text{our particle}} - \underline{s}}{||\underline{r}_{\text{our particle}} - \underline{s}||^3} \qquad \text{group's mass } M, \quad \text{group's center of mass } \underline{s} \tag{574}$$

Further details of the group can be resolved by higher order terms of the multipole expansion.

In the following we need to answer

- How do we resolve the potential of a group in less detail? (the simple monopole term in the force above will often be enough) When will this be accurate?
- How can derive a scheme, so that for the force on each particle, not all others have to be considered, but groups under small opening angle are only resolved as a whole (e.g. by the monopole term)?

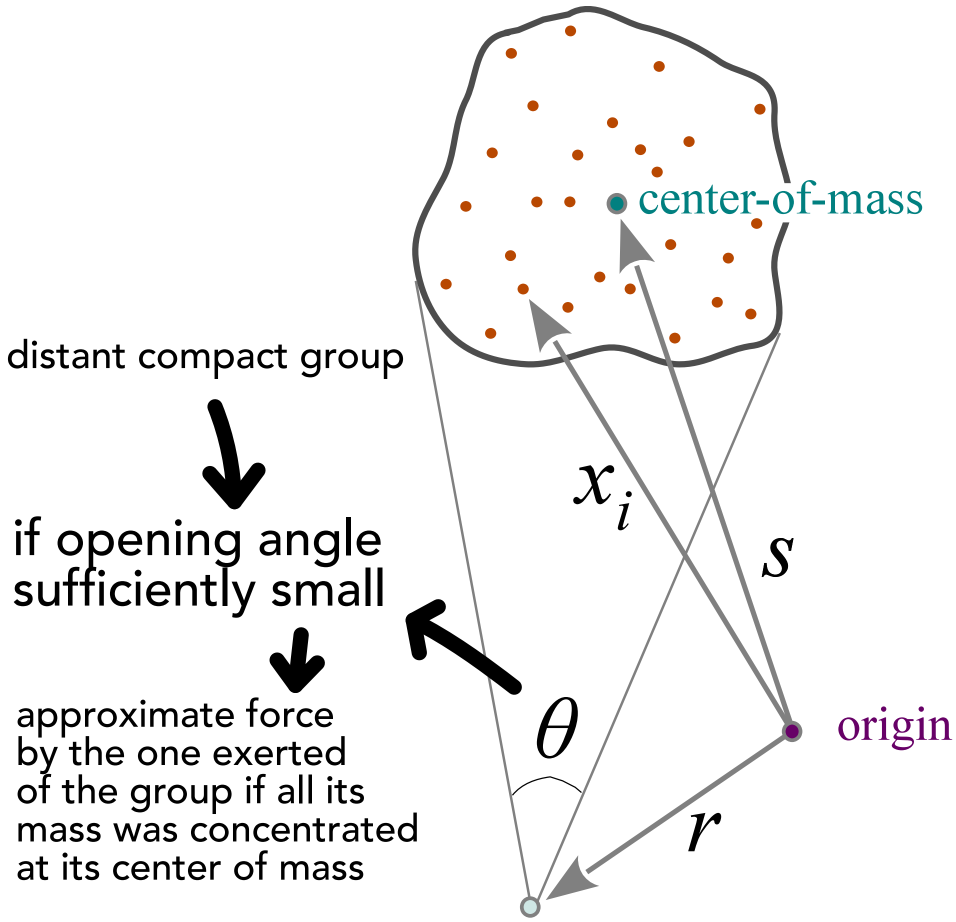


Figure 100: If the opening angle of a group is small enough, the force can be approximated by the monopole term at its center of mass.

The exact potential of the group measured at $\underline{r}$ is

$$\phi(\underline{r}) = -G \sum_i \frac{m_i}{\|\underline{r} - \underline{x}_i\|} \tag{575}$$

13.3.1 Deriving the multipole expansion

Let us rewrite the potential of the group as

$$\phi(\underline{r}) = -G \sum_i \frac{m_i}{\|\underline{r} - \underline{s} + \underline{s} - \underline{x}_i\|}, \qquad \text{group center of mass } \underline{s} = \frac{1}{M} \sum_i m_i \underline{x}_i \tag{576}$$

and define $\underline{y} := \underline{r} - \underline{s}$ and $\underline{l} = \underline{s} - \underline{x}_i$. We Taylor expand with respect to $\underline{l}$ around $\underline{l} = 0$.

Multivariate Taylor expansion:

$$f(\underline{x}+\underline{h}) = f(\underline{x}) + \underline{h}\cdot\underline{\nabla} f(\underline{x}) + \frac{1}{2}\underline{h}^T \underline{\underline{H}}_f\Big|_{\underline{x}} \underline{h} + \mathcal{O}(\underline{h}^3) \tag{577}$$

where $\underline{\underline{H}}_f$ is the Hessian matrix of f, the Jacobian of the gradient of f.

with this we get

$$f(\underline{y}+\underline{l}) = \frac{1}{\|\underline{y}+\underline{s}-\underline{x}_i\|} = \frac{1}{\|\underline{y}\|} + \frac{\underline{y}\cdot\underline{l}}{\|\underline{y}\|^3} + \frac{1}{2}\underbrace{\frac{\underline{l}^T\left[3\underline{y}\underline{y}^T - \underline{y}^2 1\right]\underline{l}}{\|\underline{y}\|^5}}_{=\frac{\underline{y}^T\left[3\underline{l}\underline{l}^T - \underline{l}^2 1\right]\underline{y}}{\|\underline{y}\|^5}} + \cdots \tag{578}$$

The numerators summed over all particles in the group lead to the *moments*

$$\begin{aligned} &\text{monopole: } M = \sum_i m_i, \qquad \text{dipole } : \underline{D} = \sum_i m_i\left(\underline{s}-\underline{x}_i\right), \\ &\text{quadrupole: } Q_{ij} = \sum_k m_k\left[3\left(\underline{s}-\underline{x}_i\right)\left(\underline{s}-\underline{x}_i\right)^T - \left(\underline{s}-\underline{x}_i\right)^2\delta_{ij}\right] \end{aligned} \tag{579}$$

Note: The gravitational dipole moment (asymmetry of the mass distribution around the center of mass) vanishes as of the definition of the center of mass.

13.3.2 Multipole expansion

Up to the quadrupole term, the potential of the group can be written as

$$\phi(\underline{r}) = -G\left(\frac{M}{\|\underline{y}\|} + \frac{1}{2}\frac{\underline{y}^T\underline{\underline{Q}}\underline{y}}{\|\underline{y}\|^5}\right), \quad \underline{y} = \underline{r}-\underline{s}, \quad \text{center of mass } \underline{s} \tag{580}$$

which we expect to be accurate if

$$\theta \underset{\text{small angle}}{\approx} \frac{\langle||\underline{x}_i - \underline{s}||\rangle}{||\underline{y}||} \approx \frac{l}{y} \ll 1, \quad \text{radius } l \text{ of the group} \tag{581}$$

so in other words if the distance of our point of interest to the center of mass of the group $\underline{y} = \underline{r} - \underline{s}$ is much larger than the radius of the group l.

13.3.3 Hierarchical grouping | baseline for smart force calculations

Let us split the particles into hierarchical groups (so a tree structure with groups and subgroups ...), which will later allow us to resolve details in the force on a particle as necessary. For all groups we calculate the multipole moments we want to consider (e.g. only the monopole, i.e. the total mass) and the center of mass.

13.3.3.1 Cosntructing the tree | Barnes and Hut oct-tree

The Barnes and Hut algorithm goes as

1. start with a cube containing all particles
2. subdivide the cube into 8 sub-cubes
3. if a cube contains more than one particle, to back to step 2 (recursion)
4. Empty sub-cubes are not stored

we therefore grow a tree with particles as leaves to purity.

In 2D we have a quad-tree (illustrated in figure 101), in 1D a binary tree.

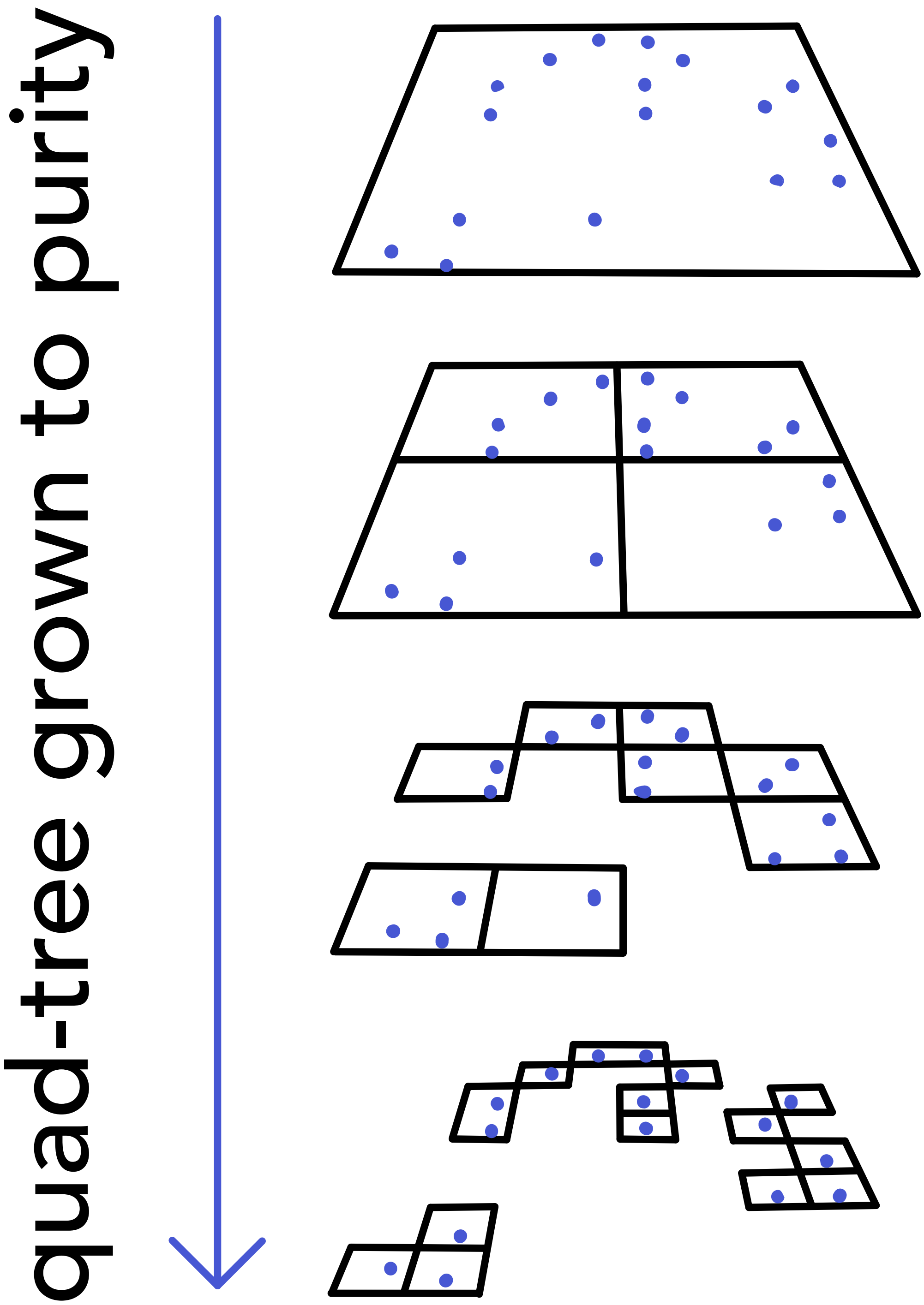


Figure 101: The Barnes and Hut algorithm constructs a tree with particles as leaves.

- Advantage - auto-adaptation: There is more refinement in dense areas and less in less dense areas.
- Disadvantage - things can go deep: The deepest level depends on the smallest separation of two particles - things go very deep if two particles are very close to each other and do not naturally fall onto some border.

Tree construction and calculation of the multipole moments

1. Build the tree by sequentially adding particles. Splits into sub-nodes are done, until the particle can be placed inside an empty sub-node.
2. Recursively compute the multipole moments
   - Does the node have sub-nodes?
     - **Yes:** Compute the multipole moments of the sub-nodes, then calculate the node's-moments based on them. For the monopole, we just have to sum the masses of the subnodes go get the monopole moment, the center of mass is the weighted average of the subnodes' centers of mass.
     - **No:** The moments should be trivial (monopole: just the particle mass, center of mass is the particle position).

Alternative grouping to the Barnes and Hut algorithm - kd-trees Alternatively, binary subdivisions along the axis can be done to do splits as illustrated in figure 102. The splitting can e.g. be done to balance fraction of mass. Advantage: The depth of the tree can easily be controlled; disadvantage: we have to store all the splitting planes - complicated bookkeeping.

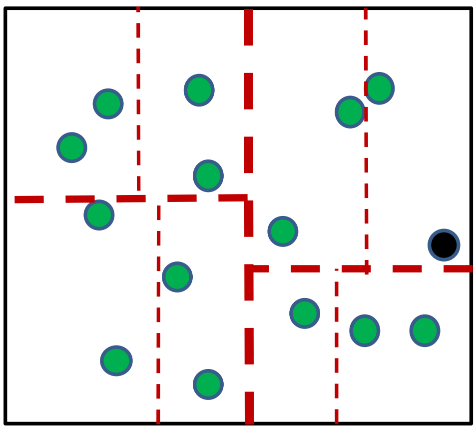


Figure 102: The kd-tree - binary subdivisions along the axes.

13.3.4 Tree walk - force calculation with adaptive group resolution

We want to calculate the force on each of our N particles, without considering all $N-1$ other particles (in total $\mathcal{O}(N^2)$ operations).

Based on the hierarchical grouping and calculated multipole moments, we can reach $\mathcal{O}(N \log N)$ (derivation follows). The idea is to for each particle calculate an approximate force acting on it using the tree-walk algorithm.

13.3.4.1 Tree walk algorithm

For each of the N particles, to calculate the force we starting at the root of the tree check for the opening angles of the sub-groups, as illustrated in figure 103.

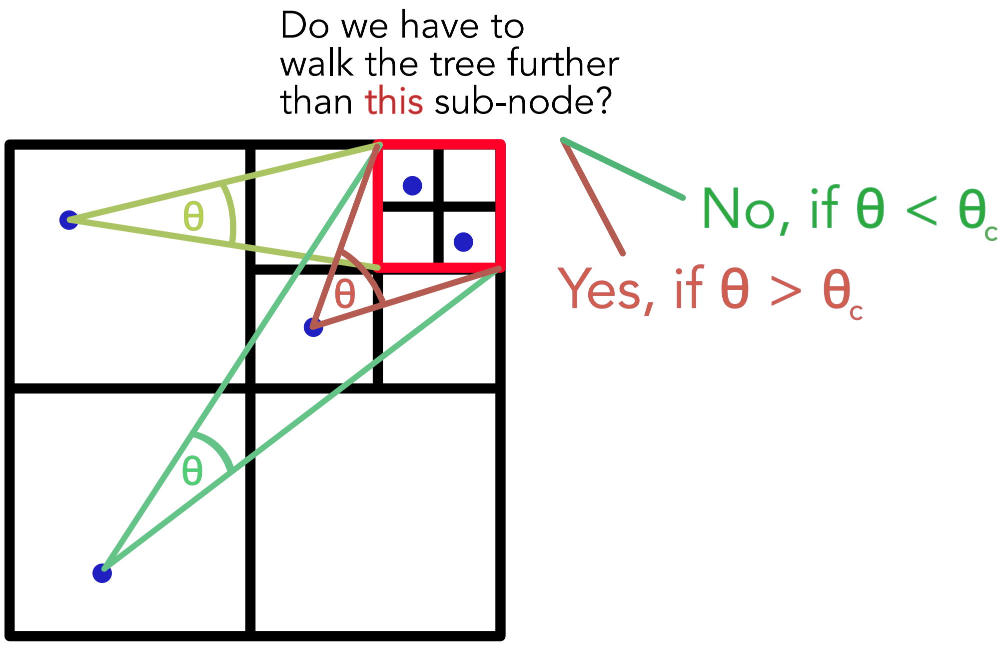


Figure 103: When do we walk further down the tree?

For each node consider the opening angle θ

- $\theta < \theta_c$ (smaller than critical angle): calculate the force from this node at our point of interest based on the multipole (or simply monopole) expansion. This is added to the total force at the point of interest.
- $\theta > \theta_c$ (larger than critical angle): open the subnodes of this node recurse

Error control by θ_c: The smaller θ_c the more expensive but also exact the force calculation. For $\theta_c = 0$ we are back at direct summation.

Fast multipole methods In our current method we calculate an approximate force for each of our N particles and then forward them based on this. In a further approximation, we can use, that the potential of a distant group feld by nearby particles will be similar - this is cheaper but also harder to parallelize.

13.3.4.2 Derivation of the cost of the tree based force calculation | $\mathcal{O}(N \log N)$

Consider N homogeneously distributed particles in a sphere with radius R. Now consider a particle at the center of the sphere. At a distance r from it, the node-size which we have so resolve in the force calculation can be estimated as

$$l^3(r), \quad l(r) \approx \theta_c r \tag{582}$$

So taking the mean particle distance

$$d = \left[\frac{\left(\frac{4\pi}{3}\right) R^3}{N}\right]^{1/3} \tag{583}$$

as the minimum distante at which other particles have to be considered, we can estimate

$$\begin{aligned} &\text{\# nodes to be considered in the force calculation} \\ &\text{for one particle } \approx \int_d^R \frac{4\pi r^2 dr}{l^3(r)} = \frac{4\pi}{\theta_c^3} \ln\frac{R}{d} \propto \frac{\ln N}{\theta_c^3} \end{aligned} \tag{584}$$

> As we have N particles with $\sim \log N$ interactions each, the total cost of the force calculation is $\mathcal{O}(N \log N)$, much better than the $\mathcal{O}(N^2)$ of direct summation.

13.3.4.3 Expected typical force errors in a monopole approximation | $(\Delta F_{\text{tot}}) \propto \theta_c^7$

We estimate the force error when resolving a group only based on the monopole (so as if all mass was concentrated at the center of mass) by the difference to the situation where all mass would be concentrated at the groups edge (group radius l).

$$\Delta F_{\text{node}} \sim |F_c - F_x| = \left|\frac{GM_{\text{node}}}{r^2} - \frac{GM_{\text{node}}}{r^2 + l^2}\right| \underset{\frac{1}{1+x} \approx 1 - x x \ll 1}{\approx} \frac{GM_{\text{node}}}{r^2}\frac{l^2}{r^2} = \frac{GM_{\text{node}}}{r^2}\theta_c^2 \tag{585}$$

plugging in $M_{\text{node}} = \frac{M}{N_{\text{nodes}}}$ with $N_{\text{nodes}} \propto \theta_c^3$ (as previously found), and summing over all nodes (akin to how variance adds up in the random walk), we get

total squared error in the force calculation for one particle based on monopole approximations:

$$\Delta F_{\text{tot}} \sim N_{\text{nodes}} \Delta F_{\text{node}} \propto \theta_c^7 \to |F_{\text{tot}}| \propto \theta_c^{3.5} \tag{586}$$

so roughly inversely proportional to the computational cost for this particle (as $N_{\text{nodes}} \propto$

θ_c^{-3}).

The smaller the opening angle, the higher the cost, the smaller the error.

13.4 Faster method II | Particle mesh technique for efficiently computing long-ranged forces

Consider for instance a plasma. Coulomb interactions are short-ranged, occuring on the scale of the Debye length (as of the shielding effect of charge) while gravity is long-ranged.

Problem: Direct summation is $\mathcal{O}(N^2)$ and all interactions are considered. However, for the short ranged interactions of a particle, only a few others in the vicinity are important and the long range interactions can safely be approximated.

Idea: Split the potential into one part describing short-ranged interactions and one part describing long-ranged interactions.

$$V = V^{\text{short}} + V^{\text{long}} \tag{587}$$

Calculate V^{short} using direct summation and V^{long} using the **particle mesh technique**.

The central idea of the particle mesh method is to use an auxiliary mesh on which the potential can be quickly calculated based on the methods discussed for the Poisson equation (multigrid relaxation or Fourier techniques).

13.4.1 Schematic particle mesh algorithm

1. Construct the density field ρ on the mesh based on the particle positions
2. Compute the potential on the mesh by solving the Poisson equation
3. Calculate the force field (on the mesh) from the discrete differentiation of the potential
4. Interpolate the forces onto the particles, move the particles appropriately and go back to step 1

We will now discuss each step.

13.4.2 Mass / charge assignment of particles to mesh cells

Setup Consider N particles $m_i, \underline{x}_i, i = 1, \ldots, N$. We assume a cubical domain with side-length L and N_g grid cells per dimension so a uniform spacing $h = \frac{L}{N_g}$. The cell centers are

$\underline{x}_{\underline{p}}$, integer index $\underline{p} = (p_x, p_y, p_z)^T, 0 \leq p_i < N_g$.

Question How should we assign the mass of the particles to the mesh cells? An intuition is given in figure 104.

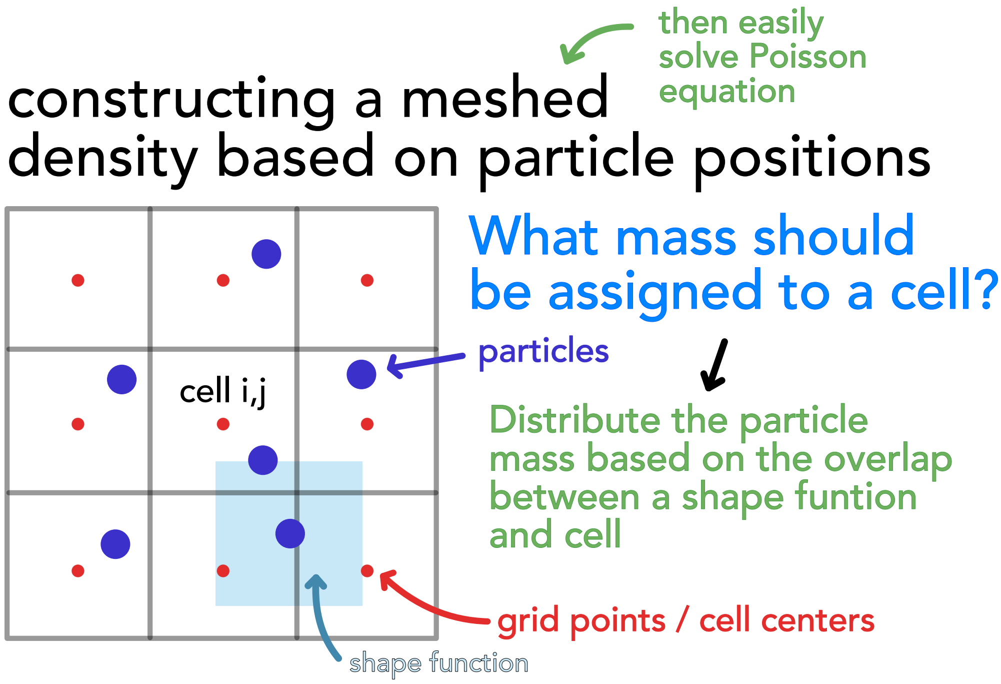


Figure 104: Mass assignment to the mesh cells.

Particle description Particles are not considered as point masses but described by normalized shape functions $S(\underline{x})$.

$$\int S(\underline{x}) d\underline{x} = 1 \tag{588}$$

Mass assignment Mass is assigned based on the overlap of a particles shape function with a mesh-cell, we assign mass of the particle to the mesh cell. The overlap over the shape function of particle i with cell with index $\underline{p}$ is

$$W_{\underline{p}}(\underline{x}_i) = \int_{\text{cell } \underline{p} \text{ so } x_p - \frac{h}{2} \text{ to } x_p + \frac{h}{2} \text{ in all dims.}} S(\underline{x}_i - \underline{x}) d\underline{x} \tag{589}$$

Using

$$\Pi(\underline{x}) = \begin{cases} 1 \text{ if } \|\underline{x}\| < \frac{1}{2} \\ 0 \text{ else} \end{cases} \tag{590}$$

we can rewrite this overlap $W_{\underline{p}}(\underline{x}_i)$ as a convolution of Π and S.

$$W_{\underline{p}}\left(\underline{x}_i\right) = \int \Pi\left(\frac{\underline{x}-\underline{x}_p}{h}\right) S\left(\underline{x}_i - \underline{x}\right) d\underline{x} \tag{591}$$

Total density of the cell with index-vector $\underline{p}$

$$\rho_{\underline{p}} = \frac{1}{h^3}\sum_{i=1}^{N} m_i W_{\underline{p}}(\underline{x}_i) \tag{592}$$

But what shape function - so what assignment scheme - should we choose?

13.4.2.1 Assignment scheme I | Nearest grid point (NGP) assignment | δ-shape function

Using $\delta(\underline{x}_i - \underline{x})$ as the shape functions

$$W_{\underline{p}}\left(\underline{x}_i\right) = \Pi\left(\frac{\underline{x}_i-\underline{x}_p}{h}\right) \tag{593}$$

so we fully assign the mass of the particle to the nearest grid point, as illustrated in figure 105.

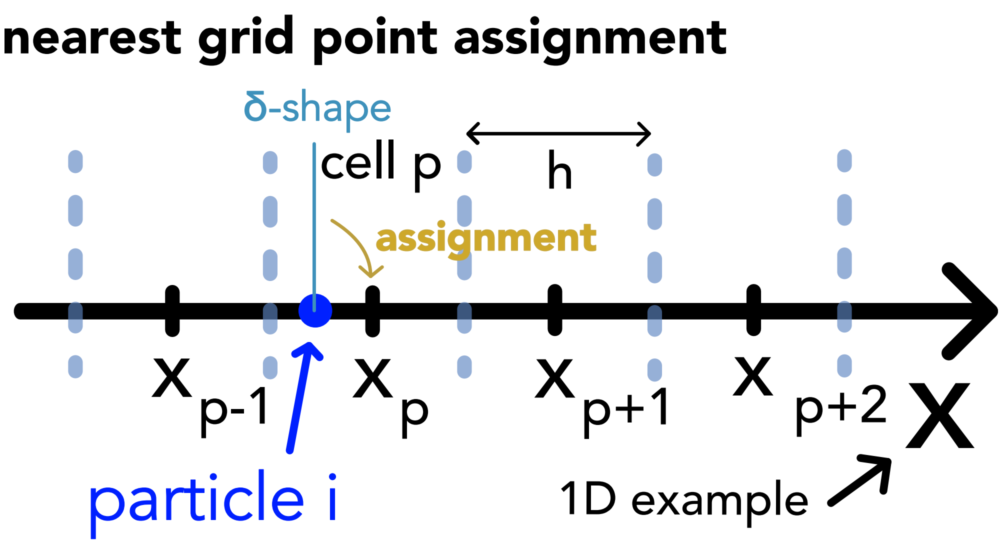


Figure 105: Nearest grid point assignment.

Problem: Mind that our particles are fiducial macro-particles making such an assignment (which would be reasonable if our particles would represent the physical ones) problematic.

13.4.2.2 Assignment scheme II | Cloud in cell (CIC) assignment | top-hat shape function

Here we use a cubical cloud shape (a top-hat)

$$S(\underline{x}) = \frac{1}{h^3}\Pi\left(\frac{\underline{x}}{h}\right), \quad \frac{1}{h^3} \text{ normalization in 3D} \tag{594}$$

of the same size as the mesh cells themselves (so at perfect overlap, all mass is assigned to the corresponding mesh cell and at max there is overlap with $2^3 = 8$ cells (in 3D)).

$$W_{\underline{p}}\left(\underline{x}_i\right) = \int \Pi\left(\frac{\underline{x} - \underline{x}_p}{h}\right) \frac{1}{h^3}\Pi\left(\frac{\underline{x}_i - \underline{x}}{h}\right) d\underline{x} \tag{595}$$

The 1D case is illustrated in figure 106 (the 2D case in figure 104)

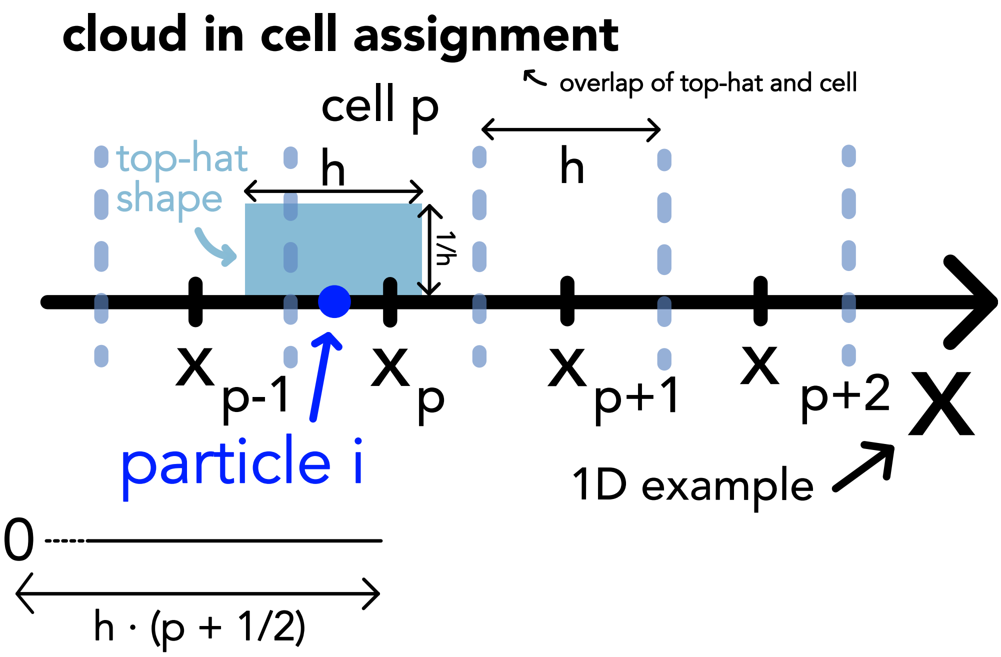


Figure 106: Cloud in cell assignment.

How to calculate the overlap practically? Consider we assign a floating point index consistent to how it is assigned to the cells, $x_i = (p_i + \frac{1}{2})h$, so $p_i = \frac{x_i}{h} - \frac{1}{2}$. The floored $\lfloor p_i \rfloor$ is the index of the cell closest to the left of that particle. The overlap with this cell is

$$W_{\lfloor p_i \rfloor}(\underline{x}_i) = 1 - (p_i - \lfloor p_i \rfloor) = 1 - p^\star, \quad W_{\lfloor p_i \rfloor + 1}(\underline{x}_i) = p^\star \tag{596}$$

(a short plausibility check is that for $p_i = \lfloor p_i \rfloor$ we have full overlap). Higher dimensions follow the same logic (but with more splits).

13.4.2.3 Assignment scheme III | Triangular shaped cloud (TSC) assignment | triangular shape function

We now use a triangular shape with a maximal overlap of 3^d cells (in d dimensions). A 1D illustration is given in figure 107.

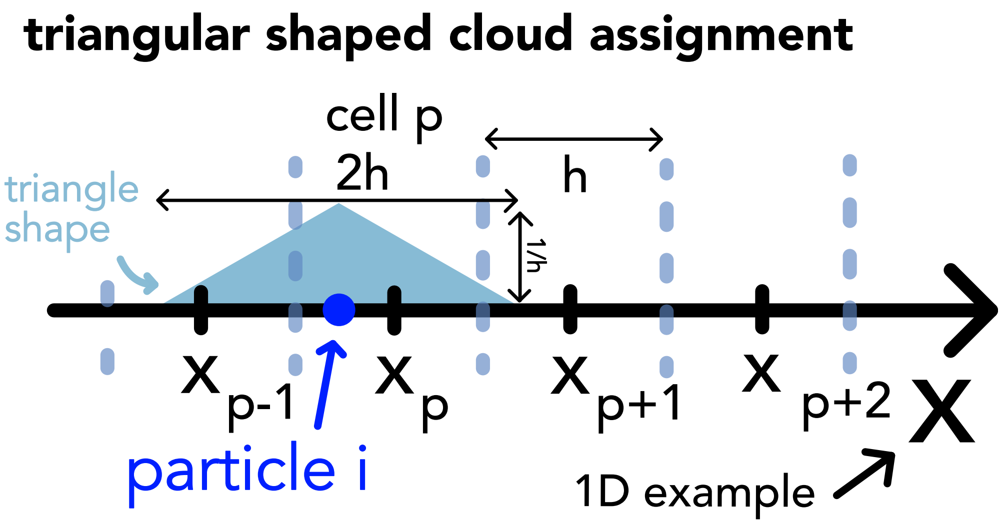


Figure 107: Triangular shaped cloud assignment.

The triangular shape can be obtained by the convolution of two top-hats, so we get

$$\begin{aligned} W_{\underline{p}}\left(\underline{x}_i\right) &= \int \Pi\left(\frac{\underline{x}-\underline{x}_p}{h}\right)\frac{1}{h^3}\Pi\left(\frac{\underline{x}_i-\underline{x}-\underline{x}'}{h}\right)\frac{1}{h^3}\Pi\left(\frac{\underline{x}'}{h}\right)d\underline{x}d\underline{x}' \\ &= \frac{1}{h^6}\int \Pi\left(\frac{\underline{x}-\underline{x}_p}{h}\right)\Pi\left(\frac{\underline{x}_i-\underline{x}}{h}\right)\Pi\left(\frac{\underline{x}'-\underline{x}}{h}\right)d\underline{x}d\underline{x}' \end{aligned} \tag{597}$$

13.4.2.4 Comparing the assignment schemes in terms of continuity

Let us consider the continuity of the force field.

- Nearest grid point (NGP) assignment: Density and hence force jump discontinuously when a particle crosses the cell boundary - at best piecewise constant force law
- Cloud-in-cell (CIC) assignment: Produces piecewise linear and continuous force but has jumps on the derivative of the force. As of the overlap information on where the particle is in the cell is stored
- Triangular shaped cloud (TSC) assignment: Here, the first derivative of the force is continuous

Name	Cloud shape $S(x)$	# of cells used	assignment function shape
NGP	$\delta(x)$	1^d	Π
CIC	$\frac{1}{h^d}\Pi\left(\frac{x}{h}\right)$	2^d	$\Pi \star \Pi$
TSC	$\frac{1}{h^d}\Pi\left(\frac{x}{h}\right) \star \frac{1}{h^d}\Pi\left(\frac{x}{h}\right)$	3^d	$\Pi \star \Pi \star \Pi$

Table 18: Overview over the assignment schemes.

The higher the order of the assignement scheme, the smoother one can make the force but the higher the computational cost (as the mass of a particle has to be distributed over more cells; more overhead in parallelization).

An overview over the assignment schemes is given in table 18.

13.4.3 Solving the Poisson equation for the potential on the mesh based on the meshed density

To solve the Poisson equation

$$\underline{\nabla}^2\phi = 4\pi G\rho \tag{598}$$

we have already discussed two methods

- Iterative solvers (relaxation methods): The solution is found based on an iterative relaxation in real space (the discretized Poisson equation is a linear system), e.g. by Jacobi or Gauss-Seidel iteration with possible speed-up by multigrid methods (making use of faster information travel / faster reduction of longwave errors on coarser grids and refinement on finer grids).
- Fourier transform based methods: The analytical solution to the potential is the convolution of the density with the Green's function of the Poisson equation. Based on the convolution theorem the convolution turns into a multiplication in Fourier space, which combined with the fast Fourier transform (FFT) is very fast ($\mathcal{O}(N\log N)$).

Assume now the potential is calculated on the mesh. **How do we calculate the force on the mesh?**

13.4.4 Calculating the force field on the mesh

Generally, the acceleration of a particle follows from the potential by

$$\underline{a} = -\underline{\nabla}\phi \tag{599}$$

which on the mesh can be done by **finite differencing**, e.g. the central difference

$$a_x(i,j,k) = -\frac{\phi(i+1,j,k) - \phi(i-1,j,k)}{2h} + \mathcal{O}\left(h^2\right), \qquad \text{cell index } \underline{p} = (i,j,k) \tag{600}$$

Using a larger stencil, higher order schemes can be constructed (based on Taylor expansion from which also the truncation error can be estimated), e.g. by the 4-point stencil

$$a_x(i,j,k) = -\frac{1}{2h}\left\{\frac{4}{3}[\Phi(i+1,j,k) - \Phi(i-1,j,k)] - \frac{1}{6}[\Phi(i+2,j,k) - \Phi(i-2,j,k)]\right\} + \mathcal{O}\left(h^4\right) \tag{601}$$

a_y and a_z follow analogously.

13.4.4.1 On the choice of the order of the finite difference scheme for the force calculation

The higher the order, the higher the accuracy, the higher the computational cost.

Note: In many collisionless systems, other errors inherent to the simulation scheme will be the bottleneck of accuracy so that the second order scheme is sufficient.

13.4.5 Interpolating the force from the mesh to the particles

We now have the forces on the mesh point but we want to know the forces on the particles.

13.4.5.1 We assign forces to the particles' positions using the same assignment kernel used to assign mass of the particles to the grid points

Idea: The mesh-nodes *give back* acceleration to a particle by the same ratio they have received mass from the particle.

Remember the mass assignment from particles to mesh cells

$$\rho_{\underline{p}} = \frac{1}{h^3}\sum_{i=1}^{N} m_i W_{\underline{p}}(\underline{x}_i) = \frac{1}{h^3}\sum_{i=1}^{N} m_i W(\underline{x}_i - \underline{x}_{\underline{p}}), \quad W_{\underline{p}}\left(\underline{x}_i\right) = \int \Pi\left(\frac{\underline{x} - \underline{x}_p}{h}\right) S\left(\underline{x}_i - \underline{x}\right) d\underline{x} \tag{602}$$

The **force interpolation** on a mass m at coordinate $\underline{x}$ is based on the acceleration field $\{\underline{a}_{\underline{p}}\}$ on the mesh, so

$$\underline{F}(\underline{x}) = m \sum_{\underline{p}} \underline{a}_{\underline{p}} W(\underline{x} - \underline{x}_{\underline{p}}) \tag{603}$$

The assignment Kernel to interpolate the forces from the mesh to the particles must be the same that was used to assign mass from the particles to the mesh cells to

- have a vanishing self-force (a particle alone on the mesh, should not start moving by ghost forces)
- have force-asymetry (pair-wise antisymmetric forces between particle pairs, Newton's third law)

Proofs of these properties follow.

13.4.5.2 Proof that for the same assignment kernel in the density and force assignment there is no self-force occuring

What is a self-force?: A self-force is a force, the particle would feel even if it was alone in the system - the particle would accelerate by itself violating conservation of momentum.

Consider the case of only one particle in the system. The force on this particle at $\underline{x}_i$ is

$$\underline{F}(\underline{x}_i) = m_i \sum_{\underline{p}} \underline{a}_{\underline{p}} W_f(\underline{x}_i - \underline{x}_{\underline{p}}), \quad \text{density assignment for } N = 1 : \rho_{\underline{p}} = m_i W_d(\underline{x}_i - \underline{x}_{\underline{p}}) \tag{604}$$

where W_f denotes a general force assignment kernel. We will see that for $W_f = W_d$ there is no self-force as of a symmetry argument.

Deriving an exact expression for $\underline{a}_{\underline{p}}$ on the grid Here we do not use the finite difference but an exact expression via the Green's function of the Poisson equation. The Greens function for the Laplace equation

$$\underline{\nabla}^2 \phi = 4\pi G \rho \tag{605}$$

fulfills

$$\underline{\nabla}^2 G(\underline{x} - \underline{x}_{\underline{p}}) = 4\pi \delta(\underline{x} - \underline{x}_{\underline{p}}) \tag{606}$$

Using this an writing the density as a combination of lots of point masses

$$\rho(\underline{x}) = \int_{-\infty}^{+\infty} \rho\left(\underline{x}_{\underline{p}}\right) \delta\left(\underline{x} - \underline{x}_{\underline{p}}\right) d\underline{x}_{\underline{p}} \tag{607}$$

we can as previously find that the potential is the convolution of the density with the Green's function

$$\begin{aligned} \underline{\nabla}^2 \phi &= 4\pi G \int_{-\infty}^{+\infty} \rho\left(\underline{x}_{\underline{p}'}\right) \delta\left(\underline{x} - \underline{x}_{\underline{p}'}\right) d\underline{x}_{\underline{p}'} \\ = \int_{-\infty}^{+\infty} \rho\left(\underline{x}_{\underline{p}'}\right) \underline{\nabla}^2 g\left(\underline{x} - \underline{x}_{\underline{p}'}\right) d\underline{x}_{\underline{p}'} &= \underline{\nabla}^2 \int_{-\infty}^{+\infty} \rho\left(\underline{x}_{\underline{p}'}\right) g\left(\underline{x} - \underline{x}_{\underline{p}'}\right) d\underline{x}_{\underline{p}'} \\ \xrightarrow{\text{check boundary conditions}} \phi &= \int_{-\infty}^{+\infty} \rho\left(\underline{x}_{\underline{p}'}\right) g\left(\underline{x} - \underline{x}_{\underline{p}'}\right) d\underline{x}_{\underline{p}'} \end{aligned} \tag{608}$$

Discretizing to our grid (sum over all grid points) we get

$$\begin{aligned} &\boxed{\underline{a}_{\underline{p}} = -\underline{\nabla}\phi_{\underline{p}} = \sum_{\underline{p}'} \underline{d}\left(\underline{p}, \underline{p}'\right) h^3 \rho_{\underline{p}'}}, \qquad \text{mass in mesh cell: } h^3 \rho_{\underline{p}'} \\ &\text{derivative of the Green's function:} \quad \underline{d}\left(\underline{p}, \underline{p}'\right) = -4\pi G \underline{\nabla} g\left(\underline{x}_{\underline{p}} - \underline{x}_{\underline{p}'}\right) \end{aligned} \tag{609}$$

where the derivative of the Greens function is antisymmetric $\underline{d}(\underline{p}, \underline{p}') = -\underline{d}(\underline{p}', \underline{p})$ so that this exact expression for the force is antisymmetric.

A symmetry argument for the absence of self-force Let us plug $\underline{a}_{\underline{p}}$ into the force expression

$$\begin{aligned} F_{\text{self}}\left(\underline{x}_i\right) &= m_i \sum_{\underline{p}} W_f\left(\underline{x}_i - \underline{x}_{\underline{p}}\right) \underline{a}_{\underline{p}} \\ &= m_i^2 \sum_{\underline{p},\underline{p}'} \underbrace{\underline{d}\left(\underline{p}, \underline{p}'\right)}_{\underline{d}(\underline{p},\underline{p}')=-\underline{d}(\underline{p}',\underline{p})} \underbrace{W_f\left(\underline{x}_i - \underline{x}_{\underline{p}}\right) W_d\left(\underline{x}_i - \underline{x}_{\underline{p}'}\right)}_{\substack{:=\mathcal{W}_{\underline{x}_i}(\underline{x}_{\underline{p}},\underline{x}_{\underline{p}'}) \text{ with} \\ \mathcal{W}_{\underline{x}_i}(\underline{x}_{\underline{p}},\underline{x}_{\underline{p}'})=\mathcal{W}_{\underline{x}_i}(\underline{x}_{\underline{p}'},\underline{x}_{\underline{p}}) \text{ for } W_f=W_d}} \\ &= 0 \text{ for } W_f = W_d \end{aligned} \tag{610}$$

as the sum over this in total antisymmetric expression vanishes.

13.4.5.3 Proof that for the same assignment kernel in the density and force assignment, the forces between particle paris are pair-wise antisymmetric

> As of Newton's third law, two particles should exert opposite but equal in magnitude forces on each other otherwise conservation of momentum would be violated.

Consider a system of two particles 1 and 2. As of the vanishing self-force the force exerted on particle 1 only follows from the mass brought onto the mesh by particle 2 and vice versa.

For the force on particle 1 we have

$$\begin{aligned}\underline{F}_{12} = m_1 \underline{a}\left(\underline{x}_1\right) \\ &= m_1 \sum_{\underline{p}} \underline{a}_{\underline{p}} W_f\left(\underline{x}_i - \underline{x}_{\underline{p}}\right) \\ &= m_1 m_2 \sum_{\underline{p},\underline{p}'} \underline{d}\left(\underline{p},\underline{p}'\right) W_f\left(\underline{x}_1 - \underline{x}_{\underline{p}}\right) W_d\left(\underline{x}_2 - \underline{x}_{\underline{p}'}\right)\end{aligned} \tag{611}$$

So for $W_f = W_d$ we can see that swapping the indices of the particles as well as $\underline{p}$ and $\underline{p}'$ and using the antisymmetry of the derivative of the Green's function yields

$$\underline{F}_{12} = -\underline{F}_{21} \quad \rightarrow \quad \underline{F}_{12} + \underline{F}_{21} = 0 \tag{612}$$

14 Random Number Generation and Monte Carlo Techniques

In the following we will discuss

- How can (pseudo-) random numbers (following a given distribution) be generated?
- How can those random numbers be useful in estimation (e.g. of integrals)? - Monte Carlo techniques
- How can we sample from very complicated distributions based on a stochastic process and how can this be useful in calculating partition functions and parameter estimation? - Monte Carlo Markov Chain

14.1 Random Number Generation and Sampling

14.1.1 An intuitive introduction to sampling

Probability distributions are ubiquitous. Consider the air molecules around you - what we feel as a temperature stems from the microscopic movement of those molecules, some moving slower, some quicker, some in one, some in another direction. Actually (under ideal conditions), if I was to measure those velocities of many molecules only along one axis and do a normalized histogram of my measurements, I would see a Gaussian emerge.

Now we want to do the opposite:

Sampling: Given a probability distribution $f(x)$, we want to generate measurements x_i such that in the limit of lots or measurements their normalized histogram resembles the probability distribution $f(x)$.

For instance to simulate my previous velocity measurement. Sampling has numerous applications, e.g. in smartly approximating integrals (Monte Carlo integration).

14.1.2 Random Number Generators - Base of all sampling methods: Sampling from the uniform distribution is *easy*

The standard continuous uniform distribution is given by the probability density function (PDF)

$$f(x) = \begin{cases} 1 & \text{if } 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{613}$$

We would now like to generate pseudo-random numbers $U \sim \mathcal{U}(0,1)$. To do so, **pseudo-random number generators** usually deterministically generate a sequence of integers which is then converted to a float in $[0,1]$.

14.1.2.1 Advantages of Pseudo-Random Number Generators over True Random Number Generators

While we can also use true random numbers (e.g. based on radioactive decay or lava lamps), pseudo-random number generators have the advantages

- they are deterministic, so the number sequence is repeatable $\rightarrow$ reproducibility, debugging
- they are usually faster
- the distribution quality is possibly better
- the distribution is not dependent on environmental factors

14.1.2.2 Desirable properties of good pseudo-random number generators

- Repeatability: Same seed $\rightarrow$ same sequence
- Randomness: the random numbers should
  - be uniformly and homogeneously distributed in $[0,1]$
  - be independent of each other (no correlation, not fully possible)
- Efficiency: Fast and memory efficient generation
- Portability: Same results across different machines
- Long period: The sequence of random numbers should at least not repeat for sufficiently long
- Insensitivity so seed: The seed should not change the characteristics of the random number distribution

14.1.2.3 A simple class of pseudo-random number generators: Linear Congruential Generators

Linear congruential generators generally follow the form

$$\begin{gathered} v_{n+1} = (a v_n + b) \mod m \in [0, m-1], \quad \text{seed } v_0 \in \mathbb{N} \\ \text{pseudo-random number } u_n = \frac{v_n}{m} \in [0,1) \text{parameters } a, b, m \in \mathbb{N} \end{gathered} \tag{614}$$

Note: As of the modulo operation, the period is at most m, where the cycle length depends on the parameters and seed, see figure 108.

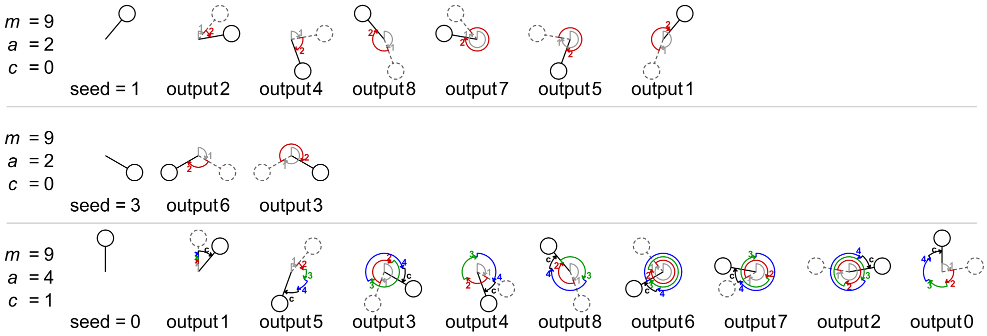


Figure 108: The cycle length of a linear congruential generator depends on the parameters and seed.

Examples of LCGs are ANSI-C, RAND, drand48 and NAG (NAG with period of 2^{48}).

While LCGs are fast and require minimum storage, but they show regular patterns. For instance if you generate lots of random numbers and from them take bunches of k and plot them in a k-dimensional space, you will see they lie on at most $(k! \cdot m)^{1/k}$ parallel $k-1$-dimensional hyperplanes. Also, if m is chosen as a power of 2 (as in the subsequently discussed RANDU), the least significant bits are not very random (the least significant one has a period of at most 2).

Let us present RANDU, an infamous, simple linear congruential generator (LCG), given by

$$v_{n+1} = (65539 \cdot v_n) \mod 2^{31}, \quad \text{seed } v_0, \quad \text{pseudo-random number } u_n = \frac{v_n}{2^{31}} \tag{615}$$

where based on an integer sequence uniformly distributed numbers u_n are generated. Now RANDU is not infamous because it is especially good, but rather because it is especially bad in the sense that generated numbers are closely related, as illustrated in figure 109.

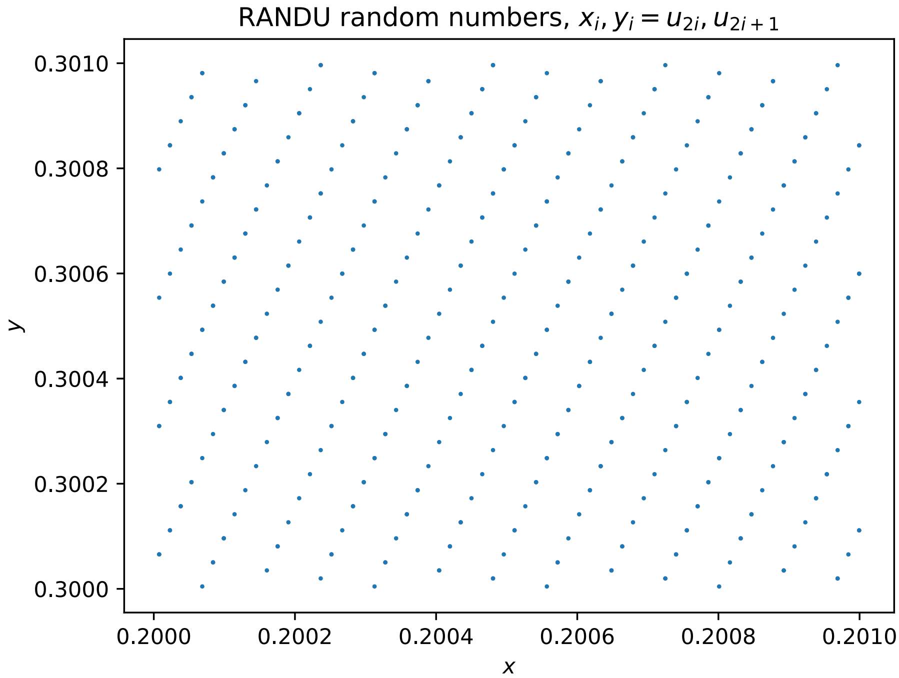


Figure 109: Consecutive numbers generated by RANDU plotted in 2D lie on parallel lines.

A more in-depth discussion about the problems of RANDU and linear congruential generators in general can be found in Press et al., 2007, chapter 7, where more advanced methods are also discussed. In R for instance the default number generator is the Marsenne-Twister (Matsumoto and Nishimura, 1998) according to the Comprehensive R Archive Network.

14.1.2.4 A first improvement | Combining multiple LCGs

$$\left.\begin{aligned} X_{i+1} &= (40014 X_i) \bmod 2147483563 \\ Y_{i+1} &= (40692 Y_i) \bmod 2147483399 \\ Z_{i+1} &= (X_i + Y_i) \bmod 2147483563 \end{aligned}\right\} \rightarrow \text{ then map } Z_i \text{ to floating point number} \tag{616}$$

14.1.2.5 Lagged Fibonacci Generators

Inspired by the Fibonacci series, we combine *lagged numbers*, i.e. numbers earlier in the series by offsets p and q

$$v_i = (v_{i-p} \odot v_{i-q}) \mod m \tag{617}$$

where $\odot$ is some arithmetic operation like addition or bitwise XOR. For instance the Mersenne Twister is based on such a method and has period $2^{19937} - 1$.

14.1.2.6 Roughly evenly spaced sampling - blue noise

Randomly sampled points in 2D will not be evenly spread - if we want more even sampling, *blue noise* generated for instance by Fast Poisson Disk sampling (sampled points are at least r apart, $\mathcal{O}(N)$, N sampled points) can be used.

Blue noise and usual randomly sampled numbers are shown in table 19. For instance the photoreceptors on our retina are laid out in a blue noise fashion.

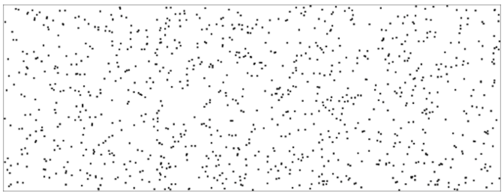
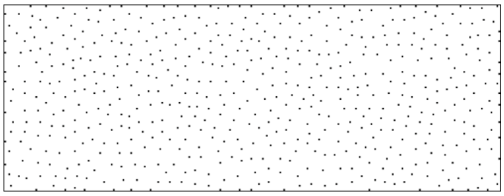
Random numbers using math.random()	Blue noise using Poisson Disk Sampling
	

Table 19: Randomly sampled points in 2D (left) and blue noise (right).

14.1.3 Inverse Transform Method | sampling from distributions with algebraically invertible cumulative distribution functions (CDFs)

Consider we want to sample from a distribution with PDF $f(x)$ and cumulative distribution function (CDF) $F(x) = \int_{-\infty}^{x} f(x')dx'$. Given a uniformly distributed variable $U \sim \mathcal{U}(0,1)$, $F^{-1}(U)$ is distributed according to f, as

$$P(F^{-1}(U) \leq x) = P(U \leq F(x)) = F(x), \quad x \in \mathbb{R} \tag{618}$$

where we used that as of $f(x) \geq 0$, $F(x)$ is monotonically increasing, so its application keeps the inequality intact.

This is the foundation of the inverse transform method which is implemented in R in code-snippet 5 and illustrated at the hand of sampling from a Gaussian in figure 110. Note that the inverse transform method would actually not be used on a Gaussian as the inverse of its CDF has no closed-form expression and e.g. the Box-Muller method (Box and Muller, 1958) would be used.

Problem: Not all CDFs are algebraically invertible.

Idea: Numerical inversion is just swapping the axes - but we have discrete inverted point with different spacing. One idea is to linearly interpolate in the to find F^{-1} at the uniformly distributed points.

```r
inverse_transform_method <- function(n, F_inv) {
    # Draw n samples from a distribution with CDF F
    # given its inverse F_inv.
    u <- runif(n) # draw n samples from U(0,1)
    return(F_inv(u)) # apply the inverse transform
}
```

Code-Snippet 5: Inverse Transform Method in R

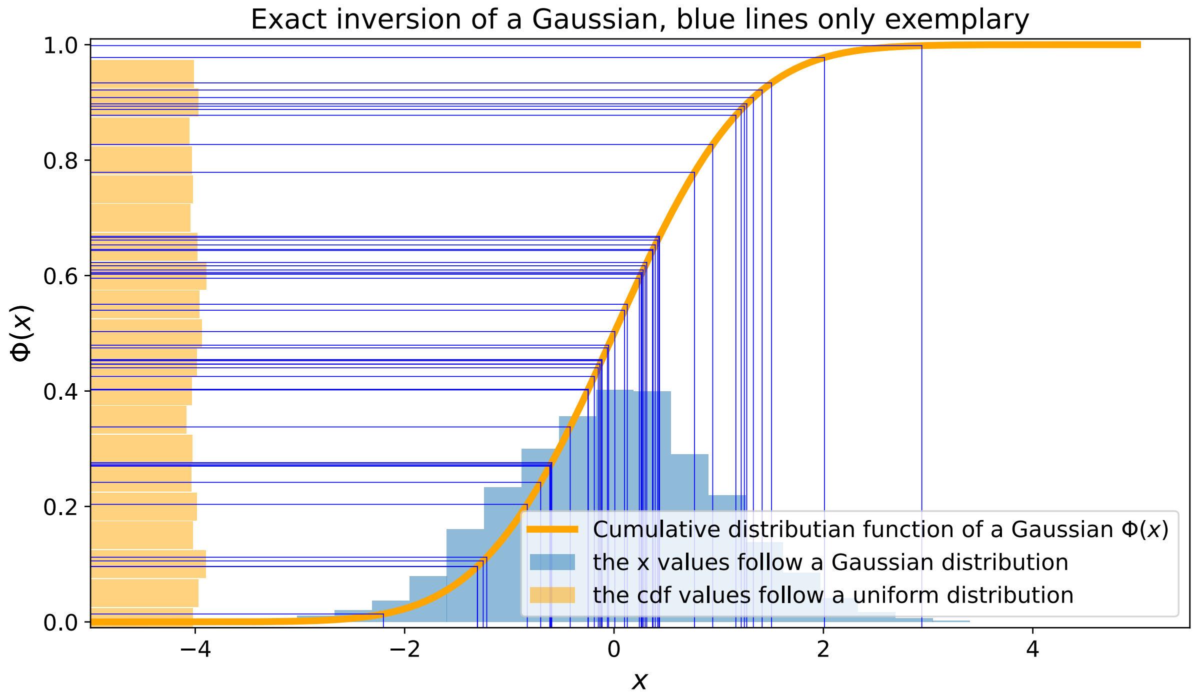


Figure 110: Illustration of the Inverse Transform Method

14.1.3.1 Alternative derivation from the transformation between probability distributions*

Consider probability distributions $f_1(x), f_2(y)$ with $y = y(x)$. Conservation of total probability implies

$$f_1(x)dx = f_2(y)dy \quad \rightarrow \quad f_2(y) = f_1(x)\left|\frac{dx}{dy}\right| \tag{619}$$

so the cumulative distribution functions are equal (?)

$$F_1(x) = \int_{-\infty}^{x} F_1(\tilde{x})d\tilde{x} = \int_{-\infty}^{y} f_2(\tilde{y})d\tilde{y} = F_2(y(x)) \tag{620}$$

so taking $f_1(x)$ as the uniform distribution, $F_1(x) = x$ and we get to the same result.

14.1.3.2 Example: Inverse Transform Method applied to the standard Laplace distribution*

The standard Laplace distribution is given by the PDF

$$f(x) = \frac{1}{2} e^{-|x|}, \quad x \in \mathbb{R} \tag{621}$$

with the CDF following from $F(x) = \int_{-\infty}^{x} f(x')dx'$ to

$$F(x) = \begin{cases} \frac{1}{2} e^{x} & \text{if } x \leq 0 \\ 1 - \frac{1}{2} e^{-x} & \text{if } x > 0 \end{cases} \tag{622}$$

with the inverse CDF

$$F^{-1}(u) = \begin{cases} \ln(2u) & \text{if } u \leq \frac{1}{2} \\ -\ln(2(1-u)) & \text{if } u > \frac{1}{2} \end{cases} \tag{623}$$

where $\frac{1}{2}$ as the transitioning point between the pieces follows intuitively from the symmetry of the PDF around $x = 0$.

The results are illustrated in figure 111.

14.1.3.3 Sampling from a Gaussian using exact inversion | Box-Muller trick

We want to sample from the Gaussian

$$f(y) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{y^2}{2}\right) \tag{624}$$

Problem: The CDF is the error function, which has no closed-form inverse.

Idea: In the Box-Muller trick, a 2D Gaussian in polar coordinates is inverted.

Consider the 2D Gaussian

$$f(x, y) = \frac{1}{2\pi} \exp\left(-\frac{x^2 + y^2}{2}\right) \tag{625}$$

note that we can also write this in polar coordinates $x = r\cos(\theta), y = r\sin(\theta)$ $(r^2 = x^2 + y^2)$ as the product of an independent radial and angular probability distribution

$$f(x, y)\, dx\, dy = f(\phi)\, d\phi \cdot f(r)\, dr = \frac{1}{2\pi} \exp\left(-\frac{r^2}{2}\right) r\, dr\, d\phi \tag{626}$$

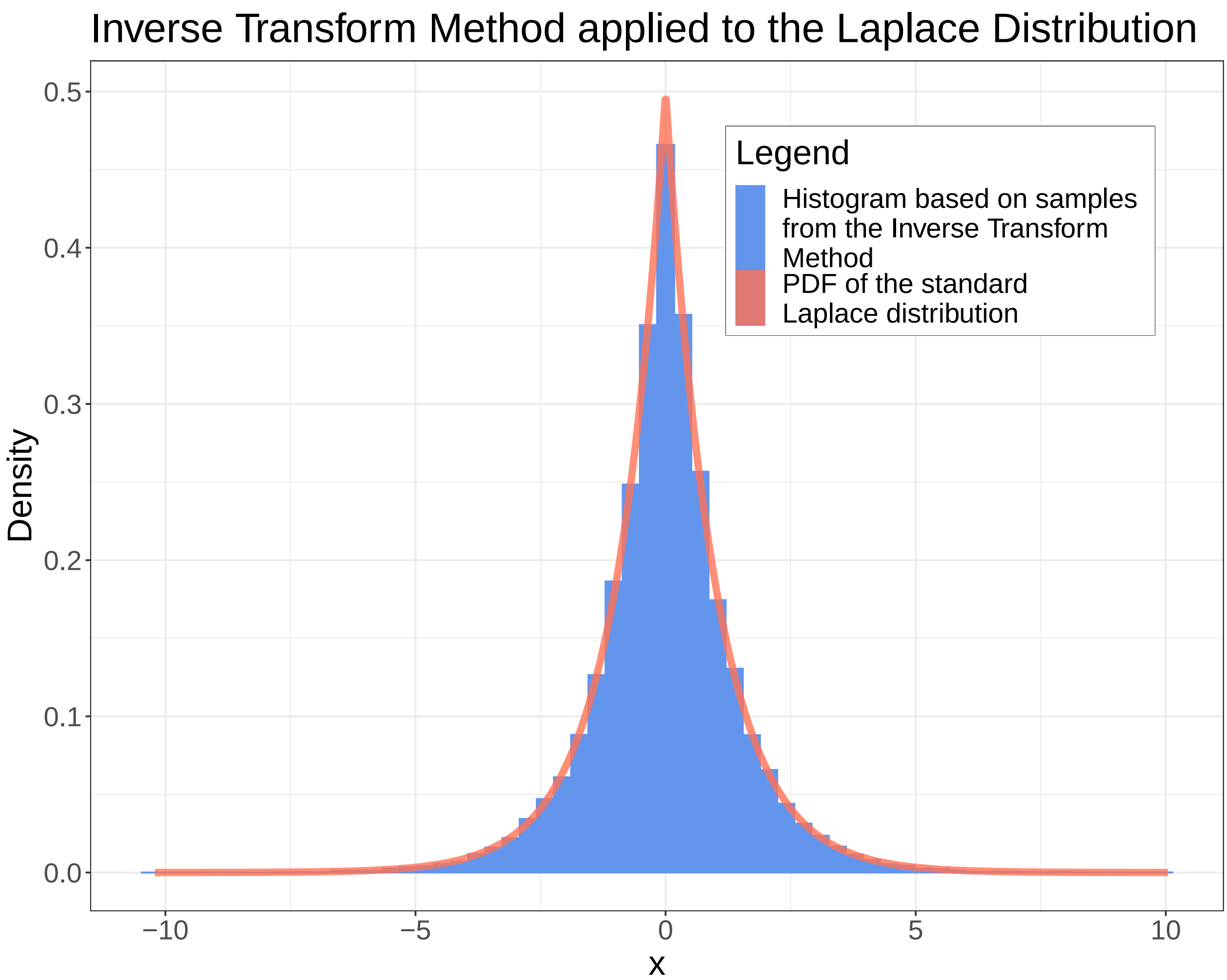


Figure 111: Illustration of the Inverse Transform Method applied to the standard Laplace distribution

where based on random numbers $u_1, u_2 \sim \mathcal{U}(0,1)$ we can sample both r and ϕ by exact inversion.

$$\begin{aligned} u_1 &= \frac{1}{2\pi}\phi \quad \rightarrow & \phi &= 2\pi u_1 \\ u_2 &= \int_0^r \exp\left(-\frac{r'^2}{2}\right) dr' = 1 - \exp\left(-\frac{r^2}{2}\right) \quad \rightarrow & r &= \sqrt{-2\ln(1-u_2)} \end{aligned} \tag{627}$$

As the 2D Gaussian is just a product of 1D Gaussians, from r and ϕ we can then get two normally distributed numbers x and y.

Box-Buller trick: Given two uniformly distributed random numbers $u_1, u_2 \sim \mathcal{U}(0,1)$, we can generate two normally distributed random numbers $x, y \sim \mathcal{N}(0,1)$ by

$$\phi = 2\pi u_1, \quad r = \sqrt{-2\ln(1-u_2)} \quad \rightarrow \quad x = r\cos(\phi), \quad y = r\sin(\phi) \tag{628}$$

14.1.3.4 Sampling from a discrete distribution*

Consider a random variable X with the following probability mass function

$$P(X=x) = \begin{cases} 0.1 & \text{if } x = 0 \\ 0.15 & \text{if } x = 1 \\ 0.25 & \text{if } x = 2 \\ 0.3 & \text{if } x = 3 \\ 0.2 & \text{if } x = 4 \\ 0 & \text{otherwise} \end{cases} \tag{629}$$

Intuitively sampling according to such a discrete distribution is simple: For uniform draws between 0 and 1, each section $a \leq x < b$ within has a probability $b - a$. Therefore, we only have to associate each x with a section of length $P(X = x)$.

The sections we are interested in are directly given by the cumulative distribution function, as illustrated in figure 112.

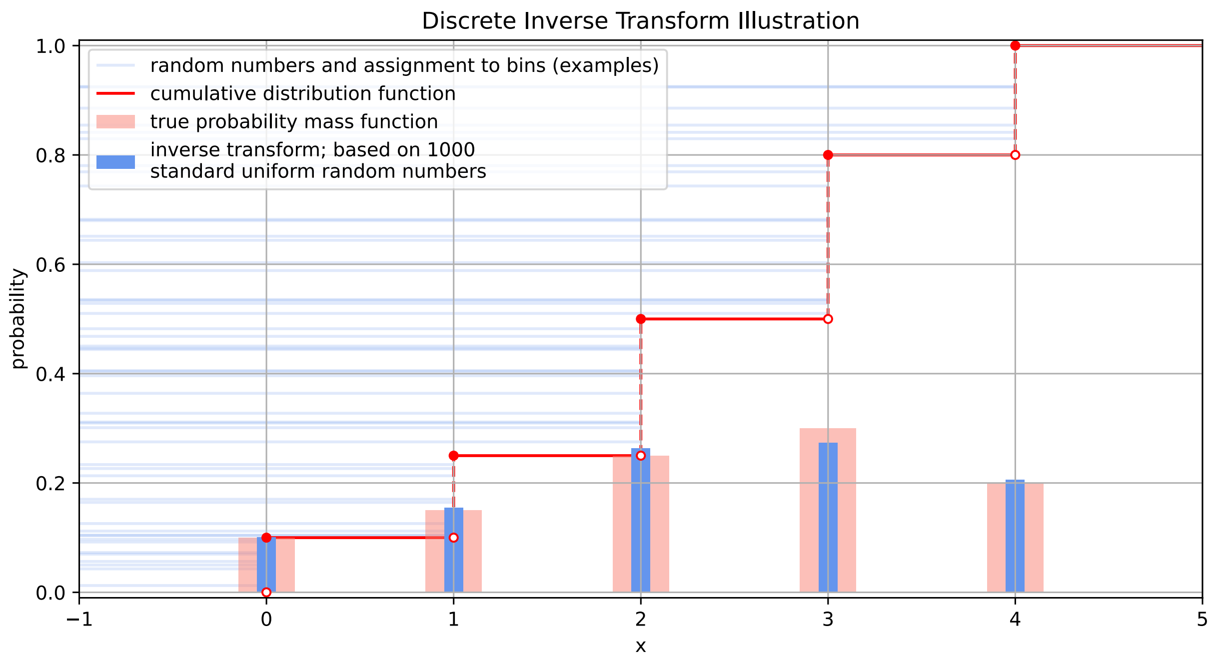


Figure 112: Illustration of the Inverse Transform Method applied to the standard Laplace distribution

14.1.4 Acceptance-rejection method

Let's say we want to sample from a complex distribution $p(x)$ and we can sample from a simple distribution $f(x)$ with $p(x) \leq Cf(x)$, some constant C.

We can then draw samples from $p(x)$ using the Acceptance-Rejection method, consisting of the following steps:

1. Generate a trial value x from $f(x)$ (e. g. using exact inversion)
2. Generate a value y from a uniform distribution $0 \leq y < Cf(x)$
3. If $y \leq p(x)$ return x as the sample value
4. Else, reject the trial value x and draw again

An illustration is provided in figure 113.

Intuition: The logic behind this is very clear if we just choose $f(x) = const.$ i. e. we uniformly sample x. This, however, would be very inefficient, so we use $Cf(x)$ that lies on top $p(x)$ as closely as possible.

Proof that we get the correct distribution: The probability dq to get (accept) a certain x within dx is

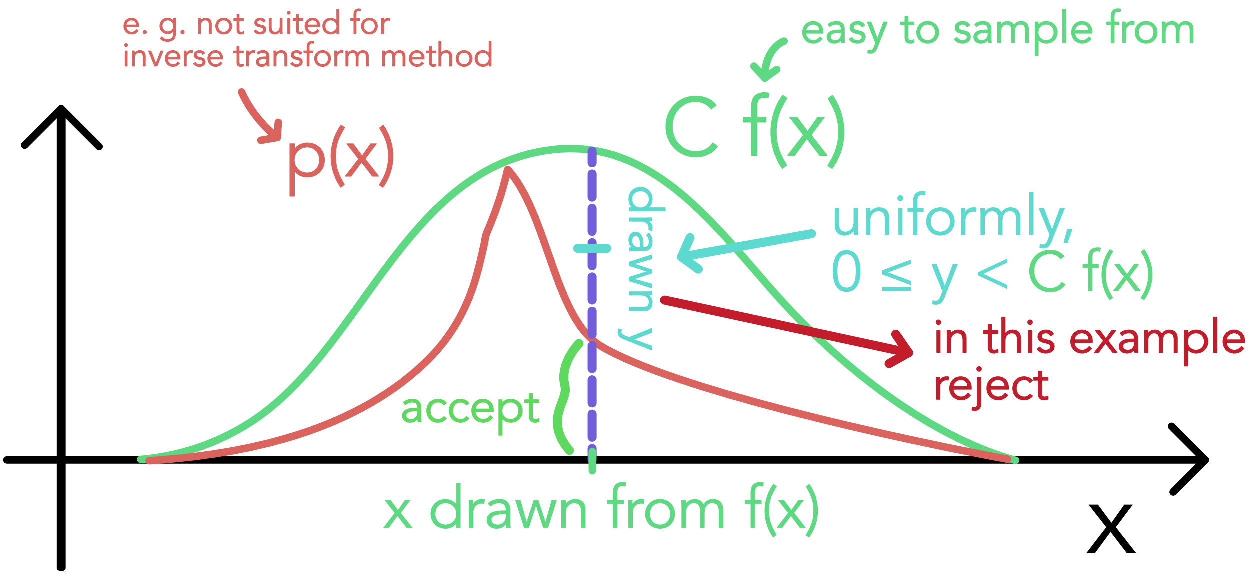


Figure 113: Illustration of the Acceptance-Rejection Method

$$dq = \underbrace{f(x)dx}_{\substack{\text{from}\\ \text{step 1}}} \underbrace{\frac{p(x)}{Cf(x)}}_{\substack{\text{step}\\ \text{2 to 4}}} = \frac{1}{C}p(x)dx \propto p(x)dx \tag{630}$$

Advantages:

- works in any dimension
- $p(\underline{x})$ does not have to be normalized (in contrast to exact inversion)

Disadvantages: Note that this is very inefficient if the rejection area is large and efficiency rapidly decreases in higher dimensions (then one might try to construct samples through a stochastic process).

14.1.4.1 Example: Sampling uniformly from a sphere

Aim: Uniformly sample points from the surface of a sphere.

In 3D we can use exact inversion to sample from the surface. The surface element in spherical coordinates is

$$dS = r^2 \sin(\theta)\, d\theta\, d\phi = r^2\, dcos(\theta)\, d\phi \tag{631}$$

(sign?) so $dcos(\theta)d\phi$ and $d\phi$ are uniform over their range. So we can use

$$\cos\theta = 1 - 2u_1, \quad \phi = 2\pi u_2, \quad u_1, u_2 \sim \mathcal{U}(0,1) \tag{632}$$

so we can sample from the surface of a sphere by

$$\begin{aligned} x &= r\sin(\theta)\cos(\phi) \\ y &= r\sin(\theta)\sin(\phi) \\ z &= r\cos(\theta) \end{aligned} \tag{633}$$

Problem: Generalization to higher dimensions is not straightforward.

Idea: Use the acceptence-rejection method.

- Uniformly sample $u_1, u_2, u_3 \sim \mathcal{U}(0,1)$
- If $r_d = u_1^2 + u_2^2 + u_3^2 \leq 1$, return (u_1, u_2, u_3), else reject and draw again
- Project the points to the surface of the unit sphere by

$$x = \frac{u_1}{r} y \quad = \frac{u_2}{r} z = \frac{u_3}{r} \tag{634}$$

which can easily be extended to higher dimensions.

14.1.4.2 Example II: Sampling from a conditioned Gamma distribution*

We would like to sample from $Gamma(2,1)$ conditional on the random variable being greater than 5, i.e.

$$p(x) = \begin{cases} \frac{x\exp(-x)}{6\exp(-5)} & \text{if } x \geq 5 \\ 0 & \text{otherwise} \end{cases} \tag{635}$$

(normalized). We use the acceptance-rejection method with $f(x)$ being an exponential distribution with $\lambda = 0.5$ conditional on the random variable being greater than 5, i.e.

$$f(x) = \begin{cases} \frac{1}{2\exp\left(-\frac{5}{2}\right)}\exp(-\frac{x}{2}) & \text{if } x \geq 5 \\ 0 & \text{otherwise} \end{cases} \tag{636}$$

which we have normalized. We can draw from $f(x)$ using the inverse transform method, where we can e.g. efficiently take care of the conditioning by drawing uniformly between $\int_0^5 \frac{1}{2}\exp(-\frac{x}{2})dx = 1 - \exp\left(-\frac{5}{2}\right)$ and 1 in the inverse transform method.

The result is illustrated in figure 114.

As for both the exponential and the gamma distribution of consideration for $x \geq 5$ the PDFs

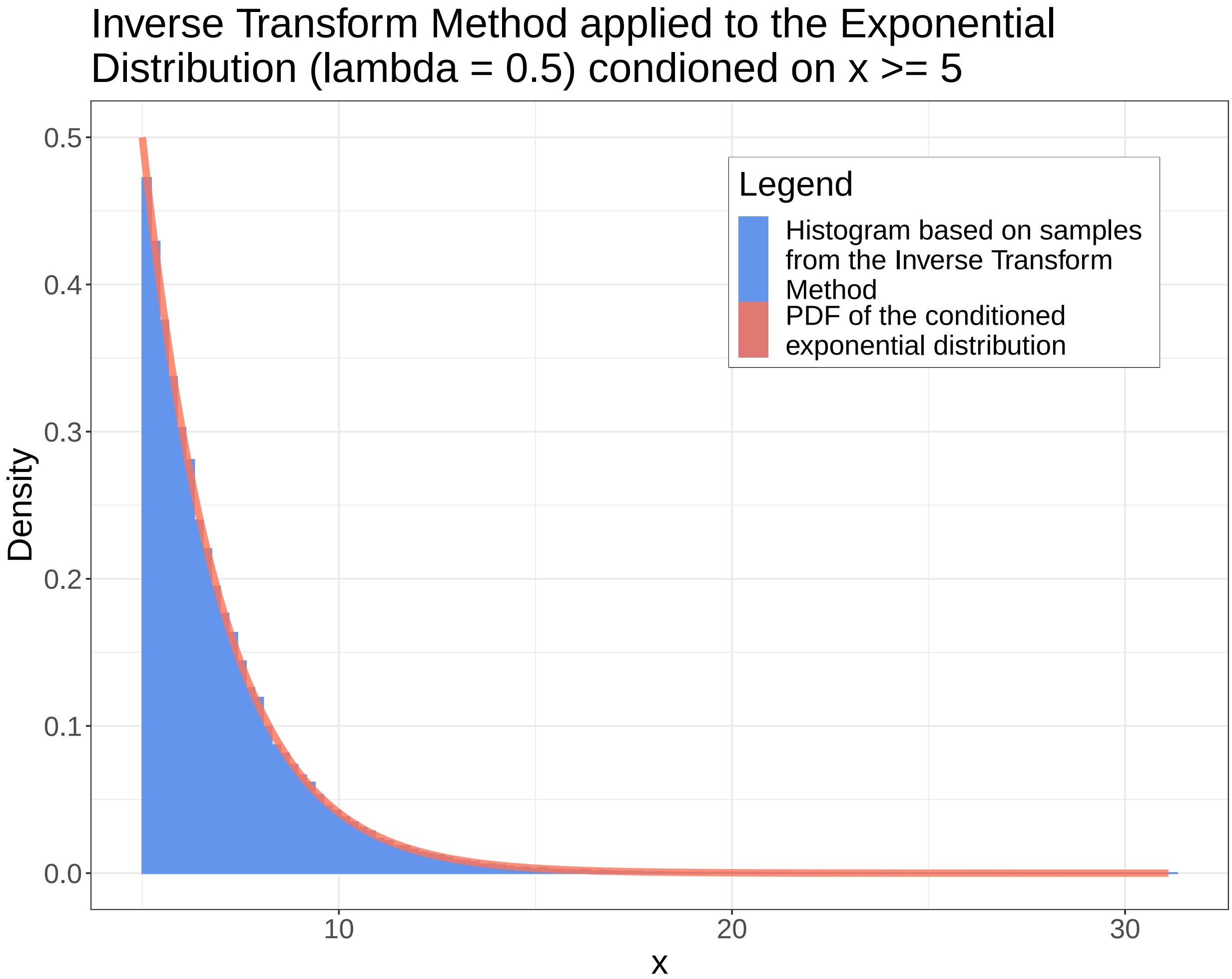


Figure 114: Application of the Inverse Transform Method to sample from a conditioned Exponential distribution

are monotonically decreasing, for C in the acceptance-rejection method, we can use

$$C = \frac{p(5)}{f(5)} = \frac{5}{3} \tag{637}$$

The results from applying the acceptance-rejection method are illustrated in figure 115.

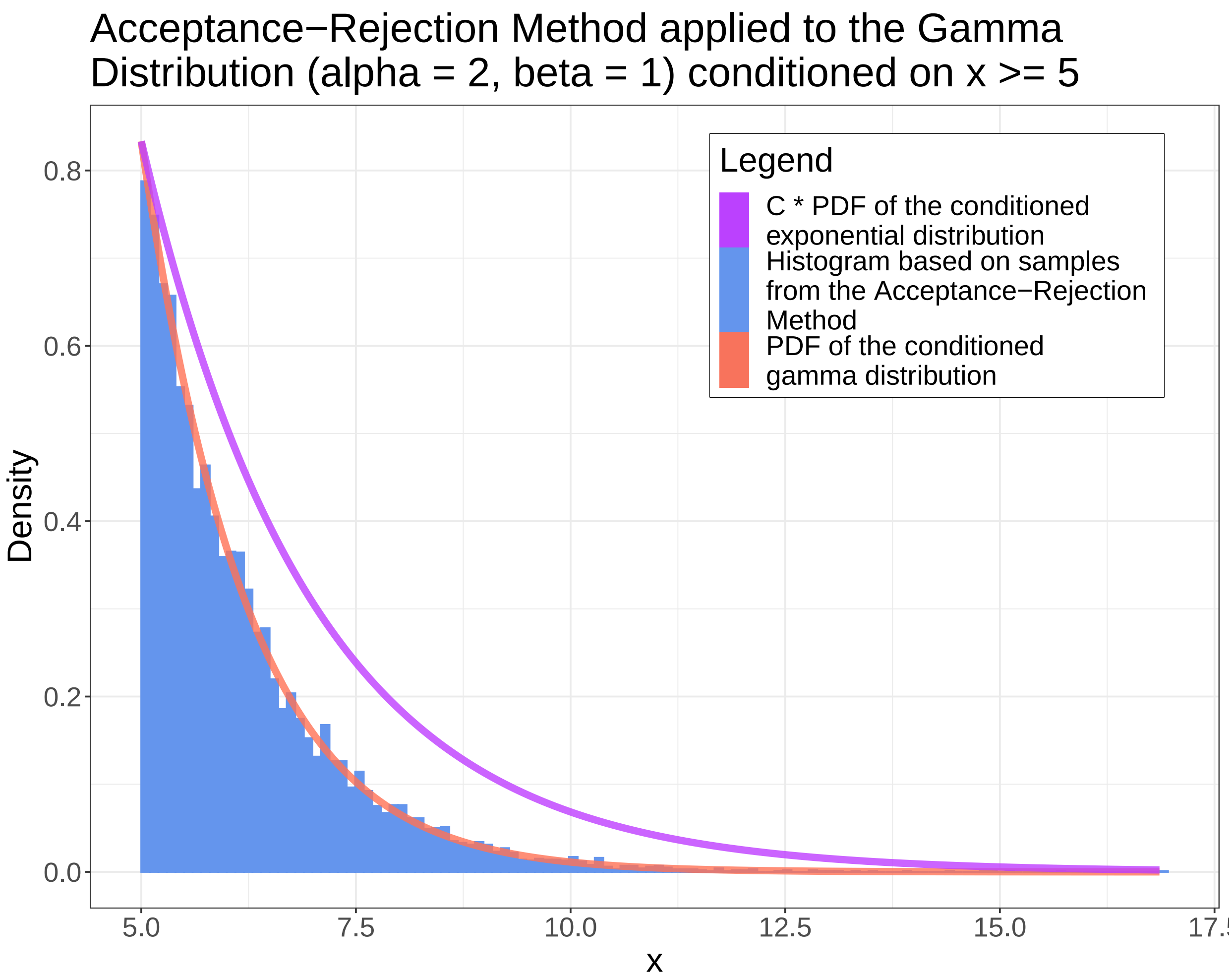


Figure 115: Application of the Acceptance-Rejection Method to sample from a conditioned Gamma distribution

14.2 Monte Carlo Estimation

14.2.1 A basic intuition for estimation

We generally draw information about reality from samples. Consider for instance we measure the energy deposited by energetic muons in a thin silicon layer. We might measure (for different muons of the same energy) $X = [0.5\,\text{MeV}, 0.6\,\text{MeV}, 1.3\,\text{MeV}]$ and we might say that in the mean the energy deposit might in general be $0.8\,\text{MeV}$ (without knowing the true distribution, from which we sampled; here quite intricate, see Particle-Data-Group et al., 2020, figure 34.7).

We have just declared our sample estimator to be at least an approximation for the population parameter - which might not be a good idea, especially for such a small sample size.

14.2.2 Monte Carlo Estimation

Monte Carlo Estimation is concerned with estimating statistics by approximating their expectations. So for a random variable with probability distribution $f(x)$ (continuous or discrete) the expectation of a function g, so

$$\theta := E[g(X)] = \begin{cases} \sum_{k=1}^{\infty} g(x_k) \cdot f(x_k) & \text{if } X \text{ is discrete} \\ \int_{-\infty}^{\infty} g(x) \cdot f(x) dx & \text{if } X \text{ is continuous} \end{cases} \tag{638}$$

is approximated by a sample mean over $(X_1, \ldots, X_m)$ (m independent realizations from the distribution of X)

$$\hat{\theta} := \frac{1}{m} \sum_{i=1}^{m} g(X_i) \tag{639}$$

which is an unbiased estimator for θ if the expectation exists and converges almost surely to θ in the limit of a large sample size (by the strong law of large numbers).

14.2.3 Distribution and Error of the Monte Carlo Estimator

Assume that $\text{Var}[g(X)] = \sigma^2 < \infty$. Then by the basic properties of the variance, we can write

$$\text{Var}[\hat{\theta}_m] = \frac{1}{m^2} \sum_{i=1}^{m} \text{Var}[g(X_i)] = \frac{\sigma^2}{m} \tag{640}$$

Notice that for the distribution of our sample mean (an addition of m i.i.d random variables) the central limit theorem applies, so

$$\frac{\hat{\theta}_m - \theta}{\sigma / \sqrt{m}} \xrightarrow[m \to \infty]{d} Z \sim \mathcal{N}(0, 1) \tag{641}$$

We would like to use the above to specify confidence intervals for our estimator $\hat{\theta}_m$. Note, however, that σ is unknown to us. Luckily, the above still holds in the limit of a large sample size when using the estimator

$$S_n^2 := \frac{1}{m-1} \sum_{i=1}^{m} (g(X_i) - \hat{\theta}_m)^2 \tag{642}$$

or rather the square root S_n. Notice that while S_n is, different from S_n^2, not an unbiased estimator, it converges in probability to σ which is sufficient to apply Slutsky's theorem yielding

$$\frac{\hat{\theta}_m - \theta}{S_n/\sqrt{m}} \xrightarrow[m\to\infty]{d} Z \sim \mathcal{N}(0,1) \tag{643}$$

14.3 Monte Carlo Integration

Classical numerical methods for integration (e.g. Simpson's rule) become infeasible for high-dimensional integrals, essentially, as the errors scale with the number of function-evaluation-points per dimension (so in high dimensions, we need to perform lots of function evaluations).

14.3.1 Intuition for Monte Carlo Integration

Consider we want to calculate the integral

$$I := \int_V g(\underline{x}) d^d \underline{x} \tag{644}$$

where V is a d-dimensional volume and g is a function $g : V \to \mathbb{R}$. Then it makes intuitive sense to approximate this integral by the mean of the function g on the domain of interest, as found as the mean of g evaluated at N random points $\underline{x}_i$ in V, multiplied by the volume of V (also denoted by V)

$$\hat{I}_N := \frac{V}{N}\sum_{i=1}^{N} g(\underline{x}_i) \xrightarrow[N\to\infty]{a.s.} I \tag{645}$$

Monte Carlo Integration: Let V for simplicity be the hypercube $[0,1]^d$ (by change of variables, the domain of interest can be mapped to this hypercube). Then Monte Carlo integration to approximate $I = \int_{[0,1]^d} g(\underline{x}) d^d\underline{x}, \underline{x} \in \mathbb{R}^d$ is given by

- Generate N random vectors $\underline{x}$ where each component is drawn uniformly from $[0,1]$ (so we need $N \times d$ random numbers)
- Evaluate
$$\hat{I}_N = V\frac{1}{N}\sum_{i=1}^{N} g(\underline{x}_i), \quad \text{here } V = 1 \tag{646}$$

Scaling of the error: The error of the Monte Carlo integrator scales with $1/\sqrt{N}$, independent of the number of dimensions of the integration d.

14.3.2 Connection to the Monte Carlo Estimator

We can also obtain the result from above from the Monte Carlo Estimator (eq. 638) by using the uniform distribution

$$f(\underline{x}) = \begin{cases} \frac{1}{V} & \text{if } \underline{x} \in V \\ 0 & \text{else} \end{cases} \tag{647}$$

and rewriting the integral as

$$I = \int_V g(\underline{x}) d^d\underline{x} = V \int_V g(\underline{x}) \cdot \frac{1}{V} d^d\underline{x} = V \int_V g(\underline{x}) \cdot f(\underline{x}) d^d\underline{x} \tag{648}$$

14.3.3 Intuition from calculating the area of a shape

Consider an irregular shape with area A_{shape} within a rectangle with area A_{rect}. The probability that a point drawn uniformly from the rectangle lies within the shape is

$$p_{\text{shape}} = \frac{A_{\text{shape}}}{A_{\text{rect}}} \tag{649}$$

so we can approximate

$$A_{\text{shape}} \approxeq A_{\text{rect}} \frac{\#\text{ hits in shape}}{\#\text{ hits in rectangle}} \tag{650}$$

which is the same as the Monte Carlo Integration introduced, where $g(\underline{x}) = 1_{\text{shape}}(\underline{x})$ and $V = A_{\text{rect}}$.

14.3.4 Error in Monte Carlo Integration - scaling with $\frac{1}{\sqrt{N}}$

Based on the connection to the Monte Carlo Estimator, the variance of our estimator for the integral is given by

$$\text{Var}[\hat{I}_N] = \frac{V^2}{N}\text{Var}[g(x)], \quad \text{estimate Var}[g(x)] \text{ by } S^2_{g;N} := \frac{1}{N-1}\sum_{i=1}^{N}\left(g(x_i) - \frac{\hat{I}_N}{V}\right)^2 \tag{651}$$

So the standard error of our Monte Carlo estimator

$$\hat{I}_N = \frac{V}{N} \sum_{i=1}^{N} g(\underline{x}_i) \tag{652}$$

that is if we calculate $\hat{I}_N$ multiple times with different random samples, the standard deviation of these different estimates is

$$\boxed{\sigma_{\hat{I}_N} = V\sqrt{\frac{\sigma_g^2}{N}} \propto \frac{1}{\sqrt{N}}} \tag{653}$$

which is illustrated in figure 116. This follows directly from the basic properties of the variance

$$\text{for } X, Y \text{ independent} \quad \text{Var}[X+Y] = \text{Var}[X] + \text{Var}[Y], \quad \text{Var}[aX] = a^2 \, \text{Var}[X] \tag{654}$$

with

$$\text{Var}[\hat{I}_N] = \text{Var}\left[\frac{V}{N}\sum_{i=1}^{N} g(\underline{x}_i)\right] = \frac{V^2}{N^2}\sum_{i=1}^{N} \text{Var}[g(\underline{x}_i)] \underset{\underline{x}_i \text{ i.i.d.}}{=} \frac{V^2}{N} \text{Var}[g(\underline{x})] \tag{655}$$

14.3.5 Distribution of $\hat{I}_N$ and derivation of the central limit theorem*

The Monte Carlo Estimator is

$$\hat{I}_N = \frac{V}{N}\sum_{i=1}^{N} g(\underline{x}_i) = \frac{1}{N}\sum_{i=1}^{N} y_i = \sum_{i=1}^{N} s_i, \quad y_i = V g(\underline{x}_i), \quad s_i = \frac{y_i}{N} \tag{656}$$

by basic properties of variance and mean

$$\begin{aligned} \langle \hat{I}_N \rangle &= N\langle s \rangle = \langle y \rangle \\ \text{Var}[\hat{I}_N] &= N \, \text{Var}[s] = \frac{1}{N}\text{Var}[y] = \frac{V^2}{N}\text{Var}[g(\underline{x})] \end{aligned} \tag{657}$$

Note: $p(y)$ is the distribution of y over V along the y-axis with

$$\langle y \rangle = \int y p(y)\, dy = \int_V g(\underline{x})\, d^d\underline{x}, \quad \int p(y)\, dy = 1 \tag{658}$$

$p(y)$ **might be any distribution.**

Applying the Central Limit Theorem (eq. 401) to the Monte Carlo Estimator (assuming the moments of $p(y)$ exist and are finite), we obtain

$$\begin{aligned} P_N(\hat{I}_N) &= \frac{1}{\sqrt{2\pi \operatorname{Var}[\hat{I}_N]}} \exp\left(-\frac{(\hat{I}_N - \langle \hat{I}_N \rangle)^2}{2 \operatorname{Var}[\hat{I}_N]}\right) \\ &= \frac{\sqrt{N}}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{N(\hat{I}_N - \langle y \rangle)^2}{2\sigma_y^2}\right) \end{aligned} \tag{659}$$

independent of the shape of $p(y)$ for N sufficiently large.

Derivation of the Central Limit Theorem by Fourier Transform The central limit theorem can be derived by considering P_N in Fourier space, where the central ingredient will simply be

$$\exp x = \lim_{N\to\infty} \left(1 + \frac{x}{N}\right)^N \tag{660}$$

which is also approximately true for large N.

Distribution of the sum of independent random variables: Consider **independent** variables X, Y with PDFs f_X and f_Y. The PDF of the sum $Z = X + Y$ is naturally the convolution

$$f_Z(z) = \int_{-\infty}^{\infty} \underbrace{f_X(x) f_Y(z-x)}_{x+(z-x)=z} \, dx \tag{661}$$

Another way to write this is

$$f_Z(z) = \int f_X(x) f_Y(y) \delta(z - (x+y)) \, dx \, dy \tag{662}$$

which one can easily see is equivalent to the convolution and also makes intuitive sense - the δ-function ensures that the sum of x and y is z.

Analogously to the two-variable case, we write

$$P_N\left(\hat{I}_N\right) = \int \delta\left(\hat{I}_N - \sum_i \frac{y_i}{N}\right) p(y_1)\, p(y_2) \cdots p(y_N)\, \mathrm{d}y_1\ \mathrm{d}y_1 \cdots \mathrm{d}y_N \tag{663}$$

Based on the Fourier transform and expansion of $p(y)$

$$\begin{aligned} \hat{p}(k) &= \int p(y) \exp\left(ik(y - \langle y \rangle)\right) \mathrm{d}y \\ &\underset{\text{Series Expansion}}{=} \int p(y) \left[1 + ik(y - \langle y \rangle) - \frac{k^2}{2}(y - \langle y \rangle)^2 + \cdots\right] \mathrm{d}y \\ &= 1 - \frac{k^2}{2}\sigma_y^2 + \cdots \end{aligned} \tag{664}$$

we find the Fourier transform of P_N to be

$$\begin{aligned}
\hat{P}_N(k) &= \int P_N\left(\hat{I}_N\right)\exp\left(ik\left(I_N - \langle I_N\rangle\right)\right)\mathrm{d}I_N \\
&\underset{\langle \hat{I}_N\rangle = \langle y\rangle}{=} \int p\left(y_1\right)\cdots p\left(y_N\right)\exp\left(i\frac{k}{N}\left(y_1 - \langle y_1\rangle + y_2 - \langle y_2\rangle + \ldots + y_N - \langle y_N\rangle\right)\right)\mathrm{d}y_1\cdots\mathrm{d}y_N \\
&= \left[\hat{p}\left(\frac{k}{N}\right)\right]^N \\
&\underset{\text{series expansion of }\hat{p}}{=} \left(1 - \frac{k^2\sigma_y^2}{2N^2}\right)^N \\
&\underset{N\text{ large}}{\approxeq} \exp\left(-\frac{k^2\sigma_y^2}{2N}\right)
\end{aligned} \tag{665}$$

where the inverse Fourier transform of $\hat{P}_N(k)$

$$P_N\left(\hat{I}_N\right) = \frac{1}{2\pi}\int \exp\left(-ik\left(I_N - \langle I_N\rangle\right)\right)\hat{P}_N(k)\mathrm{d}k \tag{666}$$

yields the previously stated result for $P_N\left(\hat{I}_N\right)$[23].

14.3.6 Illustrative Example of Monte Carlo Integration

Consider the integral

$$I = \int_0^1 g(x)dx, \quad g(x) = \exp\left(-\frac{x^2}{2}\right) \tag{667}$$

Our Monte Carlo estimator for this integral is given by

$$\hat{I}_N = \frac{1}{N}\sum_{i=1}^{N} g(x_i), \quad x_i \sim \mathcal{U}(0,1) \tag{668}$$

which is illustrated in figure 116. The higher the number of sampled points N the lower the variance of our estimator; the lower the variance of g along the y axis, the lower the variance of our estimator.

[23]Insert the result for $\hat{P}_N(k)$, complete the square and use the normalization of the normal distribution or the Gaussian integral $\int \exp\left(-\alpha x^2\right)\mathrm{d}x = \sqrt{\frac{\pi}{\alpha}}$.

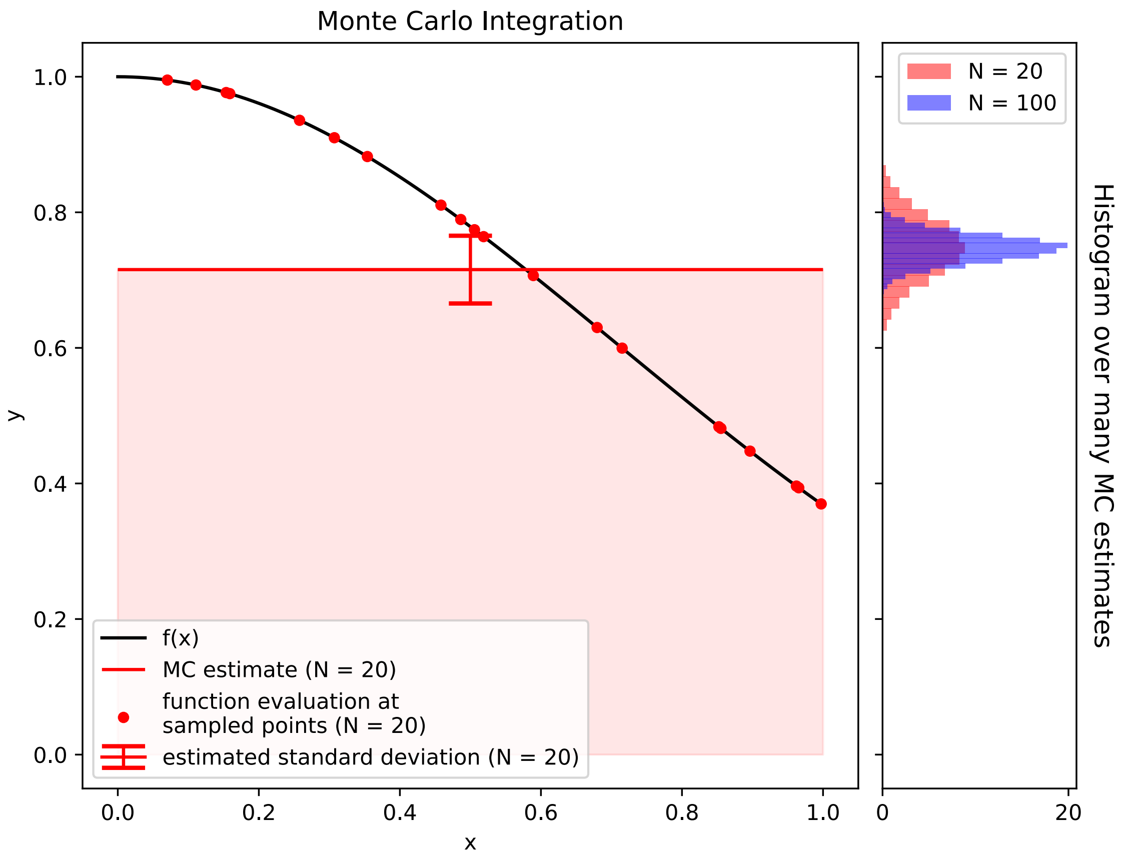


Figure 116: Illustration of Monte Carlo Integration for $I = \int_0^1 \exp\left(-\frac{x^2}{2}\right) dx$

14.3.7 Comparison to other techniques - when to use Monte Carlo integration?

Error of standard methods In a standard integration scheme like the midpoint rule or Simpsons rule[24] each dimension is divided into n regularly space points, so that we have $N = n^d$ points in total. The error scales with

$$\text{midpoint or trapezoidal rule } \propto \frac{1}{n^2}, \quad \text{Simpson's rule } \propto \frac{1}{n^4} = \frac{1}{N^{\frac{4}{d}}} \tag{669}$$

In these *regular* methods, adding more points to the integration (increasing N) helps less and less the higher we go in dimension - e.g. for Simpson's rule err $\propto \frac{1}{N^{\frac{4}{d}}}$. For a standard method if we want 10 points per dimension with $d = 10$ we already need 10^{10} function evaluations - infeasible (e.g. high-dimensional integrations in Machine Learning).

[24] $\int_a^b f(x)dx \approx \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right]$ which can be derived by approximating the function f by a quadratic polynomial and integrating this polynomial.

On the other hand in Monte Carlo integration the error scales with $\frac{1}{N^{\frac{1}{2}}}$ independent of the dimensionality d.

Starting at what dimension of the integral d will MC integration have a more favorable error scaling compared to Simpson's rule? Setting $\frac{1}{N^{\frac{4}{d}}} = \frac{1}{N^{\frac{1}{2}}}$ yields that starting at $d = 8$ Monte Carlo Integration's error will reduce more quickly if we increase N the number of points (/ function evaluations) used for the integration.

Intuition for the better scaling One intuition for this advantage of Monte Carlo is that the higher the dimension, the more evenly pairs of random points are spread with respect to their pair-wise distance - normally known as the curse of dimensionality, illustrated in figure 117.

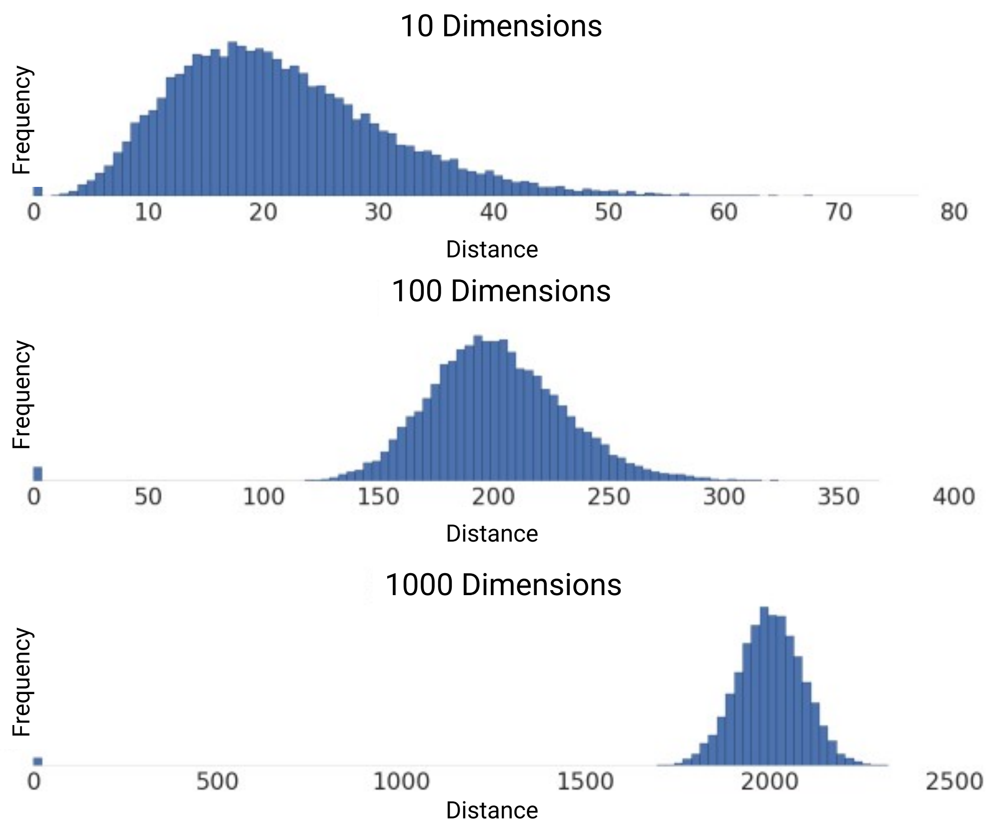


Figure 117: In higher dimensions, pairwise distances between random points show a sharper distribution around the mean distance (a more even distribution in space)

14.3.8 Reducing the variance of the Monte Carlo Estimator

14.3.8.1 Antithetic Estimators*

Two random variables with the same distribution on the same probability space are called *antithetic*, if their covariance is negative. We can use such antithetic variables to reduce the variance of a Monte Carlo estimator.

This motivates the following ansatz

$$
\begin{aligned}
I &:= \int_0^1 g(x)dx, \quad \hat{I}_{anti} = \frac{\hat{I}_{N/2} + \tilde{I}_{N/2}}{2} \\
\hat{I}_{N/2} &= \frac{1}{N/2} \sum_{i=1}^{N/2} g(u_i), \quad \tilde{I}_{N/2} = \frac{1}{N/2} \sum_{i=1}^{N/2} g(1-u_i), \quad u_i \sim \mathcal{U}(0,1)
\end{aligned}
\tag{670}
$$

with

$$
\operatorname{Var}[\hat{I}_{anti}] = \frac{1}{4}\left(\operatorname{Var}[\hat{I}_{N/2}] + \operatorname{Var}[\tilde{I}_{N/2}] + 2\operatorname{Cov}[\hat{I}_{N/2}, \tilde{I}_{N/2}]\right) \underbrace{=}_{\text{see eq. 651}} \operatorname{Var}[\hat{I}_N] - \frac{1}{2}\operatorname{Cov}[\hat{I}_{N/2}, \tilde{I}_{N/2}]
\tag{671}
$$

where for g continuous and monotonic, $\operatorname{Cov}[\hat{I}_{N/2}, \tilde{I}_{N/2}]$ is negative, as U and $1-U$ with $U \sim \mathcal{U}(0,1)$ are negatively correlated. So we can reduce the variance compared to an estimator $\hat{I}_N$ using the same number of function evaluation and only half the number of random numbers.

Let us give an intuition why using antithetic evaluation points makes sense for estimating an integral of a monotonic function from 0 to 1. Using uniform antithetic variables we create pairs of points $(x_i, 1-x_i)$, which are symmetric around $x = 0.5$ and as of the monotony of the function g we want to integrate, we obtain a better balance between sampling points where g is large and where g is small, reducing the variance of the estimate.

Example of using Antithetic Estimators

Consider the integral

$$
I = \int_0^2 \frac{1}{1+x^2} dx \underbrace{=}_{\text{exact solution}} \arctan(2) \approx 1.107149
\tag{672}
$$

Estimates based on the standard Monte Carlo estimator and the antithetic estimator are given in table 20 and illustrated in figure 118. A relative reduction in standard deviation of roughly 80% is achieved using the antithetic estimator.

Estimator	Estimate	Standard Deviation	95% Confidence Interval
Standard Monte Carlo	1.096	0.017	[1.0627 1.128]
Antithetic Monte Carlo	1.106	0.0033	[1.099 1.112]

Table 20: Rounded estimates for the integral $\int_0^2 \frac{1}{1+x^2}dx$ using the standard Monte Carlo estimator and the antithetic Monte Carlo estimator for $N = 100$ samples with seed 1234

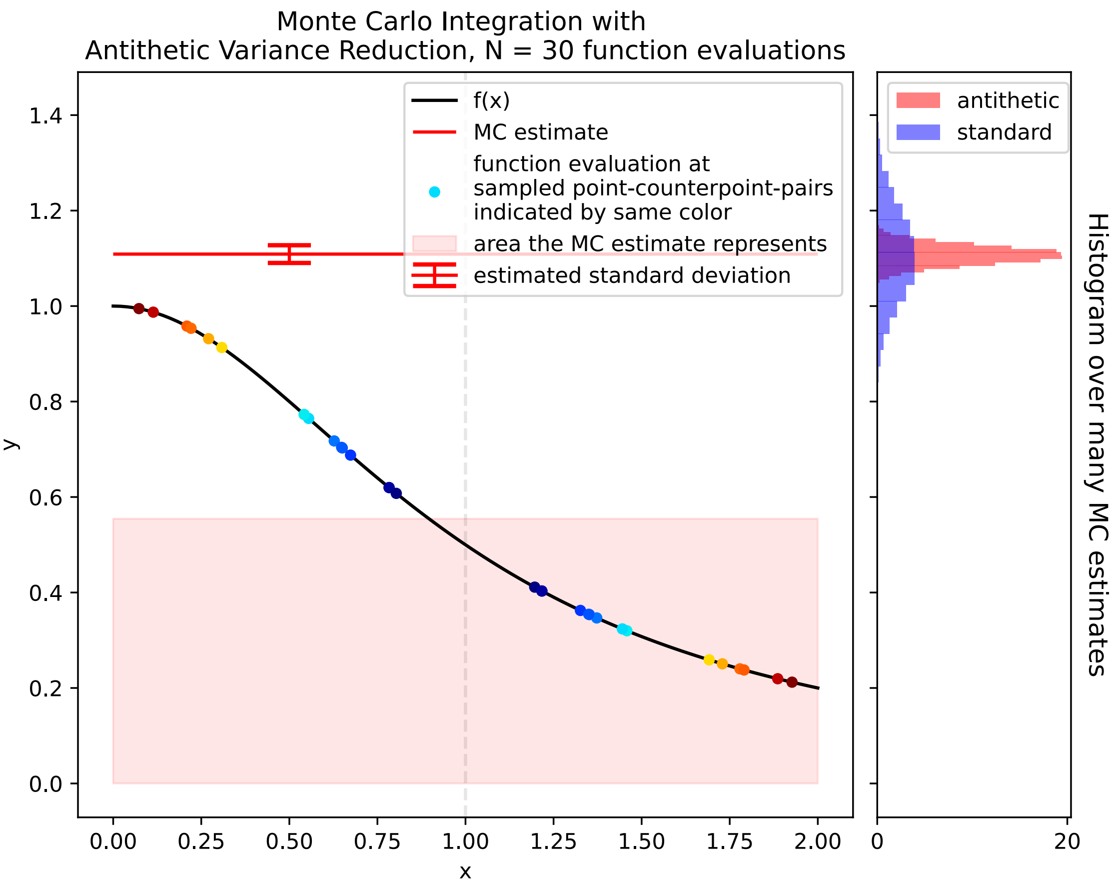


Figure 118: Illustration of Monte Carlo Integration with Antithetic Variables. Notice the even distribution of points around $\frac{a+b}{2} = 1$ by construction yielding an advantage over the standard Monte Carlo estimator.

14.3.8.2 Importance Sampling

Problem: Consider we have a sharply peaked function $g(x)$ over the domain of integration. In this case using evaluation points sampled from the uniform will be inefficient.

Importance sampling idea: Preferably choose points where the integrand is large. This is done by sampling from a distribution $f(x)$ similarly shaped as $g(x)$ but from which we can easily sample (e.g. by direct inversion[a] or the rejection method).

[a]Note that for the inverse transform we need the inverted CDF, so we need to integrate the PDF. So choosing $f(x)$ itself as $g(x)$ is non-sense as we would need to integrate $f(x)$ to obtain the CDF.

The higher the variance of the output of the function $g(x)$ over the domain of integration, the higher the variance in our Monte Carlo Estimate for our integral.

$$I := \int_V g(x)dx = \int_V \frac{g(x)}{f(x)} f(x)dx, \quad I_N = \frac{1}{N}\sum_{i=1}^{N} \frac{g(x_i)}{f(x_i)} \tag{673}$$
$$\text{sample } \{x_i\}_{i=1,\ldots,N} \text{ drawn from } f(x) \text{ limited to } V$$

Let us as an example tackle the integral

$$I = \int_0^1 \frac{4}{1+x^2} dx \underbrace{=}_{\text{exact solution}} \pi \tag{674}$$

using the importance sampling function $f(x) = \frac{4-2x}{3}$ on $0 \leq x \leq 1$. We can sample from $f(x)$ using the inverse transform method using the CDF $F(x) = \frac{4x-x^2}{3}$ (with $F(1) = 1$ as we want) so $F^{-1}(u) = 2 - \sqrt{4-3u}$. We can then simply use equation 673 to obtain an estimate.

The importance sampling approach compared to the standard Monte Carlo approach is illustrated in figure 119 and 120. There it is easy to see how effectively integrating a flatter function is to our advantage.

Advantage of Importance Sampling: As we can see in figure 119 the flatter $h(x) := \frac{g(x)}{f(x)}$ (best $h \propto g$ as achievable in lattice Monte Carlo) (mind the different naming in the figure) the sharper the probability distribution $p(h)$ (a limited range of h values with higher probabilities), the smaller the Monte Carlo error $\sigma_{\hat{I}_N} = \sqrt{\frac{\sigma_h^2}{N}}$.

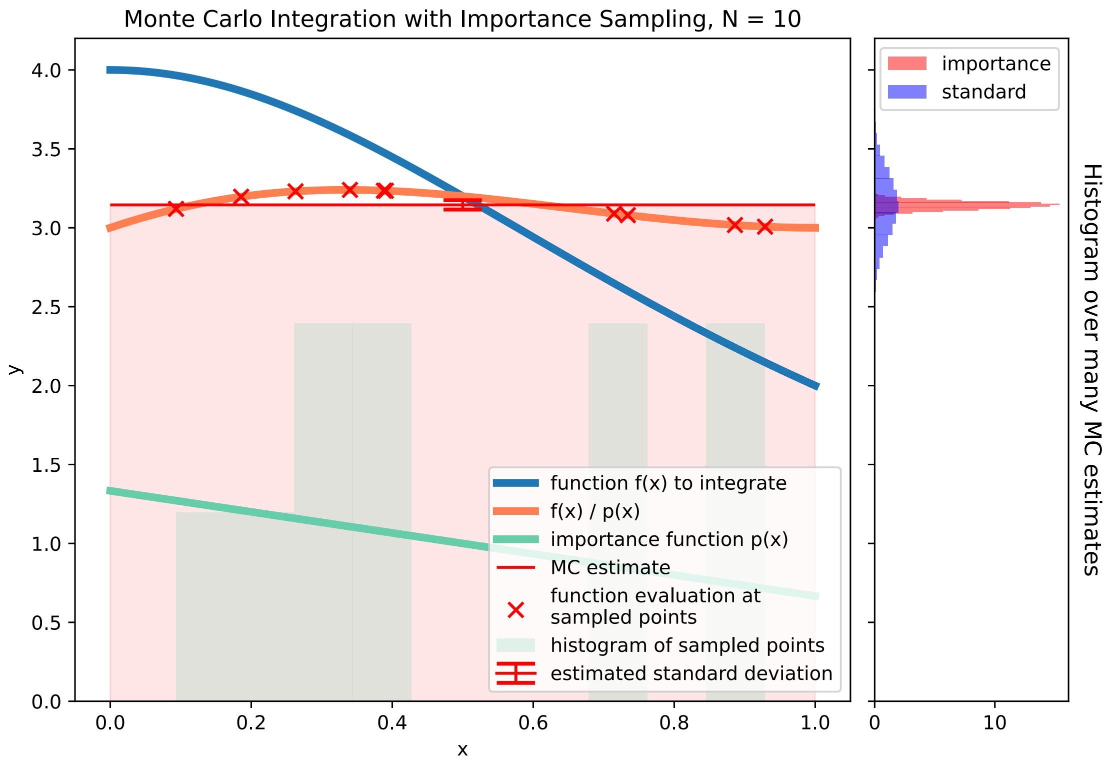


Figure 119: Illustration of Monte Carlo Integration with Importance Sampling for $N = 10$ samples.

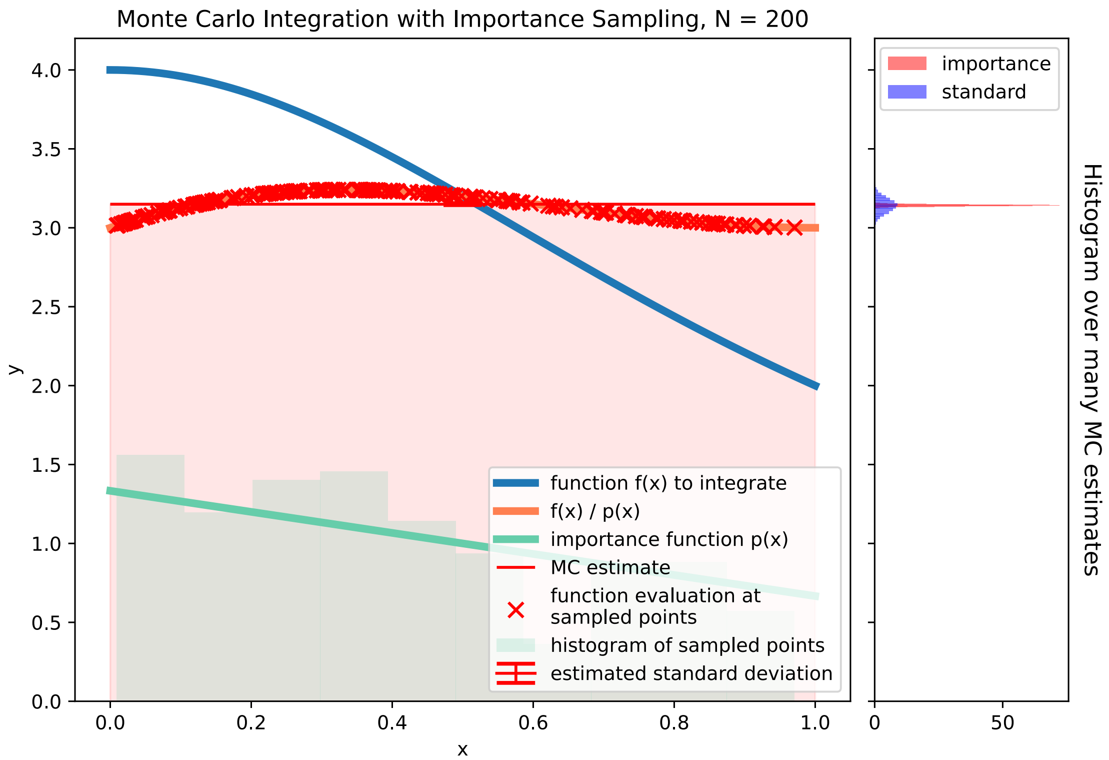


Figure 120: Illustration of Monte Carlo Integration with Importance Sampling for $N = 200$ samples.

14.4 Sampling with a stochastic process

Let us recapitulate

- We first introduced how one can sample uniformly distributed random numbers, based on generating an integer sequence (e.g. the infamously bad RANDU)
- We then introduced how one can - based on the capability of sampling from the uniform distribution - sample from other distributions. If the Cumulative Distribution Function (CDF) of a distribution is invertible, we can use the inverse transform method if not for instance the rejection method can be used.
- We have seen how uniform random numbers can be used in standard Monte Carlo integration and how the variance of the Monte Carlo estimate can be reduced by importance sampling - based on our ability to sample from a given distribution.

Problem: What can we do if neither direct inversion nor the rejection method are viable for sampling from a distribution $p(\underline{x})$, e.g. if p is complicated and $\underline{x}$ very high-dimensional?

In the following we will construct a stochastic process, a sequence of numbers where the next one is chosen stochastically, so that if we at some point take a sequence of those numbers, they are a sample of the distribution $p(\underline{x})$, so if we take a sufficiently large sequence, its histogram will approximate the distribution $p(\underline{x})$ - **the process has** $p(\underline{x}) = p_{eq}(\underline{x})$ **as its equilibrium distribution**.

14.4.1 Markov Process and Markov chain | Monte Carlo Markov Chain (MCMC)

Under conditions detailed later, a discrete sequence of states

$$x_1 \xrightarrow{f} x_2 \xrightarrow{f} x_3 \xrightarrow{f} \dots \xrightarrow{f} x_n, \quad \text{Monte-Carlo operator } f \tag{675}$$

is called Markov chain generated by a Markov process.

We will first discuss characterizing properties of Markov processes and then introduce an algorithm the Metropolis Hastings algorithm fulfilling these.

14.4.1.1 Characterizing property of the Markov process | Memorylessness

The probabilistic Monte Carlo steps f from one state of the system to the next (one number to the next) are taken with a transition probability W_f

$$W_f(x \to x') = W_f(x'|x) \tag{676}$$

which only depends on the current state - no information on the history is used.

This conditional transition property has the natural properties

$$\int W_f(x \to x')\, dx' = 1, \quad W_f(x \to x') \geq 0 \tag{677}$$

14.4.1.2 Transitioning the probability distribution

The probability of a value x' depending on the prior distribution $p(x)$ is

$$p(x) \xrightarrow{f} p'(x') = \int p(x) W_f(x \to x')\, dx \tag{678}$$

Idea: Imagine this process would have an equilibrium, fixed-point distribution $p_{eq}(x)$, then subsequent applications of f will yield us a sequence of numbers that are a sample of $p_{eq}(x)$.

14.4.1.3 Properties demanded of the update step f | stochastic process has equilibrium distribution | ergodicity

1. Equilibrium distribution: There is an equilibrium distribution $p_{eq}(x)$ that is preserved by the update step (once reached, no other distribution comes forth), i.e.

$$p_{eq}(x) = \int p_{eq}(x) W_f(x \to x')\, dx \tag{679}$$

   so $p_{eq}(x)$ is a fixed point of the update step f.

2. Ergodicity - entire domain must be reachable: Starting from any state x by repeatably applying f we must be able to get arbitrarily close to any other state x'.

14.4.1.4 Nice consequences of the demanded properties of the update step

- Ensemble approaches equilibrium distribution: Any ensemble of states (a set of numbers) approaches the equilibrium distribution $p_{eq}(x)$ if f is applied sufficiently often.
- Collection of states from a chain approaches equilibrium distribution: The collection of states $x_1, \ldots, x_n$ in a single Markiv chain under action of f approaches p_{eq} as the number of steps n goes to infinity.

We therefore have to principal ways of sampling: **ensemble sampling** (we apply f repeatedly to a set of states, which at some point are samples from p_{eq}) and **chain sampling** (starting from one initial state samples are taken as sequences from the chain).

14.4.1.5 Proof that updating the probability distribution converges to its fixed-point equilibrium distribution p_{eq}

We proof this by first showing that each update step f will bring us closer to the equilibrium distribution and secondly that there is only one fixed-point distribution p_{eq}.

$p'(x')$ is closer to $p_{eq}(x')$ than $p(x)$ as

$$\begin{aligned}
||p' - p_{eq}|| &\equiv \int |p'(x') - p_{eq}(x')|\, dx' \\
&\underset{\text{plug in } p'(x'), p_{eq}(x')}{=} \int \left| \int (p(x) - p_{eq}(x))\, W_f(x \to x')\, dx \right| dx' \\
&\underset{\text{triangle inequality}}{\leq} \int \int |p(x) - p_{eq}(x)|\, W_f(x \to x')\, dx\, dx' \\
&\underset{W_f \text{ normed}}{=} \int |p(x) - p_{eq}(x)|\, dx \\
&= ||p - p_{eq}||
\end{aligned} \tag{680}$$

so $||p' - p_{eq}|| \leq ||p - p_{eq}||$.

Why can't there be two fix-point distributions p_{eq}, p'_{eq} with $||p_{eq} - p'_{eq}|| > 0$? As of ergodicity all states are reachable from any other state. As of memorylessnes the transition only depends on the current state. If there were two fixed-point distributions, as of the above properties, nothing would prevent the system from transitioning from one fixed-point distribution to the other - and they therefore would not be fixed-point distributions in the first place. So it is really $||p' - p_{eq}|| < ||p - p_{eq}||$.

Note: Ergodicity is more easily followed by f in practice than one might imagine.

14.4.1.6 Detailed balance | Common condition for the update steps

A stronger version of p_{eq} must be a fix-point of the update step is

$$p_{eq}(x)W_f(x \to x') = p_{eq}(x')W_f(x' \to x) \quad \to \quad p_{eq} \text{ is fix-point under } f \tag{681}$$

where the looser fix-point condition (eq. 679) follows by integrating over x.

Therefore a Markov Chain fulfilling detailed balance has p_{eq} as equilibrium. In total we want W_f to fulfill detailed balance and ergodicity.

But how do we choose W_f?

14.4.2 Metropolis Hastings algorithm - simple and generic construction of the transition f

Starting from a current state x we want to go to the next one x'. How do we choose the next state?

1. x' is proposed with the proposal probability $q(x \to x')$ (fairly arbitrary, must be ergodic though)

> A simple symmetric update step could be
>
> $$\begin{gathered} q(x \to x') : x' = x + e, \quad e \text{ distributed symmetrically around } 0 \\ \text{e.g. } e \sim \mathcal{N}(0, \sigma^2) \text{ so } q(x \to x') = \mathcal{N}(x - x', \sigma^2) \end{gathered} \tag{682}$$

References

Ardourel, Vincent (2017). »Irreversibility in the derivation of the Boltzmann equation«. In: *Foundations of physics* 47.4, pages 471–489.

Biamonte, Jacob et al. (2017). »Quantum machine learning«. In: *Nature* 549.7671, pages 195–202.

Box, G. E. P. and Mervin E. Muller (1958). »A Note on the Generation of Random Normal Deviates«. In: *The Annals of Mathematical Statistics* 29.2, pages 610–611. DOI: 10.1214/aoms/1177706645. URL: https://doi.org/10.1214/aoms/1177706645.

Bryant, Randal E and David Richard O'Hallaron (2011). *Computer systems: a programmer's perspective*. Prentice Hall.

Chou, C et al. (2007). »Numerical methods for stiff reaction-diffusion systems«. In: *DISCRETE AND CONTINUOUS DYNAMICAL SYSTEMS SERIES B* 7.3, page 515.

Courant, R., K. Friedrichs, and H. Lewy (Dec. 1928). »Über die partiellen Differenzengleichungen der mathematischen Physik«. In: *Mathematische Annalen* 100.1, pages 32–74. DOI: 10.1007/BF01448839. URL: https://doi.org/10.1007/BF01448839.

Graziani, F. et al. (2022). »Shock physics in warm dense matter: A quantum hydrodynamics perspective«. In: *Contributions to Plasma Physics* 62.2, e202100170. DOI: https://doi.org/10.1002/ctpp.202100170. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/ctpp.202100170.

Hairer, Ernst (June 2006). »Long-time energy conservation of numerical integrators«. In: *Lecture Notes Ser. FoCM Santander 2005* 331. DOI: 10.1017/CBO9780511721571.005.

Hairer, Ernst, Christian Lubich, and Gerhard Wanner (2003). »Geometric numerical integration illustrated by the Störmer–Verlet method«. In: *Acta Numerica* 12, pages 399–450. DOI: 10.1017/S0962492902000144.

Hairer, Ernst and Gerhard Wanner (1996). *Solving Ordinary Differential Equations II*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-05221-7. URL: https://doi.org/10.1007/978-3-642-05221-7.

Hairer, Ernst, Gerhard Wanner, and Christian Lubich (2006). *Geometric Numerical Integration*. Springer-Verlag. DOI: 10.1007/3-540-30666-8. URL: https://doi.org/10.1007/3-540-30666-8.

Hairer, Ernst, Gerhard Wanner, and Syvert P. Nørsett (1993). *Solving Ordinary Differential Equations I*. Springer Berlin Heidelberg. DOI: 10.1007/978-3-540-78862-1. URL: https://doi.org/10.1007/978-3-540-78862-1.

Heiter, Pascal Frederik (2012). »On Numerical Methods for Stiff Ordinary Differential Equation Systems«. Master's thesis. Ulm University. URL: https://www.uni-ulm.de/

fileadmin/website_uni_ulm/mawi.inst.070/abschlussarbeiten/masterthesis_pfh.pdf.

Higham, Nicholas J. (2002). *Accuracy and Stability of Numerical Algorithms*. Second. Society for Industrial and Applied Mathematics. DOI: 10.1137/1.9780898718027. URL: https://epubs.siam.org/doi/abs/10.1137/1.9780898718027.

Kincl, Ondřej and Michal Pavelka (Mar. 2023). »Globally time-reversible fluid simulations with smoothed particle hydrodynamics«. In: *Computer Physics Communications* 284, page 108593. DOI: 10.1016/j.cpc.2022.108593. URL: http://dx.doi.org/10.1016/j.cpc.2022.108593.

Lambert, J.D. (1991). *Numerical Methods for Ordinary Differential Systems: The Initial Value Problem*. Wiley. URL: https://books.google.de/books?id=P0vPnQEACAAJ.

Lefever, R. and G. Nicolis (1971). »Chemical instabilities and sustained oscillations«. In: *Journal of Theoretical Biology* 30.2, pages 267–284. DOI: https://doi.org/10.1016/0022-5193(71)90054-3. URL: https://www.sciencedirect.com/science/article/pii/0022519371900543.

Lewis, Roland W, Perumal Nithiarasu, and Kankanhalli N Seetharamu (2004). *Fundamentals of the finite element method for heat and fluid flow*. John Wiley & Sons.

Margolin, L. G. and N. M. Lloyd-Ronning (June 2023). »Artificial viscosity—then and now«. In: *Meccanica* 58.6, pages 1039–1052. DOI: 10.1007/s11012-022-01541-5. URL: https://doi.org/10.1007/s11012-022-01541-5.

Matsumoto, Makoto and Takuji Nishimura (1998). »Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator«. In: *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 8.1, pages 3–30.

Mitchell, Nigel L., Eduard I. Vorobyov, and Gerhard Hensler (Nov. 2012). »Collisionless stellar hydrodynamics as an efficient alternative to N-body methods«. In: *Monthly Notices of the Royal Astronomical Society* 428.3, pages 2674–2687. DOI: 10.1093/mnras/sts228. URL: https://doi.org/10.1093/mnras/sts228.

Moser, Jürgen (1978). »Is the solar system stable?« In: *The Mathematical Intelligencer* 1.2, pages 65–71. DOI: 10.1007/BF03023062. URL: https://doi.org/10.1007/BF03023062.

Particle-Data-Group et al. (Aug. 2020). »Review of Particle Physics«. In: *Progress of Theoretical and Experimental Physics* 2020.8, page 083C01. DOI: 10.1093/ptep/ptaa104. URL: https://doi.org/10.1093/ptep/ptaa104.

Press, William H. et al. (2007). *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. 3rd edition. USA: Cambridge University Press.

Rackauckas, Christopher and Qing Nie (2017). »Differentialequations.jl–a performant and feature-rich ecosystem for solving differential equations in julia«. In: *Journal of Open Research Software* 5.1, page 15.

Rackauckas, Christopher, Sciemon, et al. (Nov. 2022). *SciML/SciMLBook: v1.1*. Version v1.1. DOI: `10.5281/zenodo.7347643`. URL: `https://doi.org/10.5281/zenodo.7347643`.

Rein, Hanno and David S. Spiegel (Nov. 2014). »ias15: a fast, adaptive, high-order integrator for gravitational dynamics, accurate to machine precision over a billion orbits«. In: *Monthly Notices of the Royal Astronomical Society* 446.2, pages 1424–1437. DOI: `10.1093/mnras/stu2164`. URL: `http://dx.doi.org/10.1093/mnras/stu2164`.

Springel, Volker et al. (2023). *Lecture notes in Fundamentals of Simulation Methods*.

Ulmann, Bernd (May 2020). *Analog and Hybrid Computer Programming*. De Gruyter. DOI: `10.1515/9783110662207`. URL: `http://dx.doi.org/10.1515/9783110662207`.

van Leer, Bram (1979). »Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method«. In: *Journal of Computational Physics* 32.1, pages 101–136. DOI: `https://doi.org/10.1016/0021-9991(79)90145-1`. URL: `https://www.sciencedirect.com/science/article/pii/0021999179901451`.