Hans Petter Langtangen,[*] Geir K. Pedersen[†]

Scaling of Differential Equations

May 5, 2016

Springer

[*]Center for Biomedical Computing, Simula Research Laboratory and Department of Informatics, University of Oslo.

[†]Department of Mathematics, University of Oslo.

Preface

Finding proper values of physical parameters in mathematical models is often quite a challenge. While many have gotten away with using just the mathematical symbols when doing science and engineering with pen and paper, the modern world of numerical computing requires each physical parameter to have a numerical value, otherwise one cannot get started with the computations. For example, in the simplest possible transient heat conduction simulation, a case relevant for a real physical material needs values for the heat capacity, the density, and the heat conduction coefficient of the material. In addition, relevant values must be chosen for initial and boundary temperatures as well as the size of the material. With a dimensionless mathematical model, as explained in Chapter 3.2, *no physical quantities* need to be assigned (!). Not only is this a simplification of great convenience, as one simulation is valid for any type of material, but it also actually increases the understanding of the physical problem.

Scaling of differential equations is basically a simple mathematical process, consisting of the chain rule for differentiation and some algebra. The *choice of scales*, however, is a non-trivial topic, which may cause confusion among practitioners without extensive experience with scaling. How to choose scales is unfortunately not well treated in the literature. Most of the times, authors just state scales without proper motivation. The choice of scales is highly problem-dependent and requires knowledge of the characteristic features of the solution or the physics of the problem. The present notes aim at explaining "all nuts and bolts" of the scaling technique, including choice of scales, the algebra, the interpretation of dimensionless parameters in scaled models, and how scaling impacts software for solving differential equations.

Traditionally, scaling was mainly used to identify small parameters in mathematical models, such that perturbation methods based on series expansions in terms of the small parameters could be used as an approximate solution method for differential equations. Nowadays, the greatest practical benefit of scaling is related to running numerical simulations, since scaling greatly simplifies the choice of values for the input data and makes the simulations results more widely applicable. The number of parameters in scaled models may be much less than the number of physical parameters in the original model. The parameters in scaled models are also dimensionless and express *ratios* of physical effects rather than levels of individual effects. Setting meaningful values of a few dimensionless numbers is much easier than determining physically relevant values for the original physical parameters.

Another great benefit of scaling is the physical insight that follows from dimensionless parameters. Since physical effects enter the problem through a few dimensionless groups, one can from these groups see how different effects compete in their impact on the solution. Ideally, a good physical understanding should provide the same insight, but it is not always easy to “think right” and realize how spatial and temporal scales interact with physical parameters. This interaction becomes clear through the dimensionless numbers, and such numbers are therefore a great help, especially for students, in developing a correct physical understanding.

Since we have a special focus on scaling related to numerical simulations, the notes contain a lot of examples on how to program with dimensionless differential equation models. Most numerical models feature quantities with dimension, so we show in particular how to utilize such existing models to solve the equations in the associated scaled model.

Scaling is not a universal mathematical technique as the details depend on the problem at hand. We therefore present scaling in a range of specific applications, starting with simple ODEs, progressing with basic PDEs, before attacking more complicated models, especially from fluid mechanics.

Chapter 1 discusses units and how to make programs that can automatically take care of unit conversion (the most frequent mathematical mistake in industry and science?). Section 2.1 introduces the mathematics of scaling and the thinking about scales in a simple ODE problem modeling exponential decay. The ideas are generalized to nonlinear ODEs and to systems of ODEs. Another ODE example, on mechanical vibrations, is treated in Section 2.2, where we cover many different physical contexts and different choices of scales. Scaling the standard, linear wave equation is the topic of Chapter 3.1, with discussion of how boundary and initial conditions influence the choice of scales. Another PDE example, the diffusion equation, appears in Chapter 3.2. Here we progress from a simple linear diffusion equation in 1D to a study of how scales are influenced by an oscillatory boundary condition. Nonlinear diffusion models, as well as convection-diffusion PDEs, are elaborated on. The final Chapter is devoted to many famous PDEs arising from continuum models: elasticity, viscous fluid flow, thermal convection, etc.

The mathematics is translated into complete computer codes for the ODE and simpler PDE problems.

Experimental fluid mechanics is a field full of relations involving dimensionless numbers such as the Grashof and Prandtl numbers, but none of the textbooks the authors have seen explain how these numbers actually relate to dimensionless forms of the governing equations. Consequently, this non-trivial topic is particularly highlighted in the fluid mechanics examples.

The mathematics in the first two chapters is very gentle and requires no more background than basic one-variable calculus and preferably some knowledge of differential equation models. The next chapter involves PDEs and assumes familiarity with basic models for wave phenomena, diffusion, and combined convection-diffusion. The final chapter is meant for readers with knowledge of the physics and mathematics of continuum mechanical models. The mathematical level of the text rises quickly after the first two chapters.

In the first two chapters, much of the mathematics is accompanied by complete (yet short) computer codes. The programming level requires familiarity with procedural programming in Python. As the mathematical level rises, the computer codes get much more comprehensive, and we refer to some files for computational examples in chapter three.

The pedagogy is to saturate the reader with lots of detailed examples to provide an understanding for the topic, primarily because the choice of scales depends on the problem at hand. One can also view the notes as a reference on how to scale many of the most important differential equation models in physics. For the simpler differential equations in Chapters 2 and 3, we present computer code for many computational examples, but the treatment of the advanced models in Chapter 4 is more superficial to limit the size of that chapter.

The exercises are named either Exercise or Problem. The latter is a stand-alone exercise without reference to the rest of the text, while the former typically extends a topic in the text or refers to sections or formulas in the text.

What this booklet is and is not

Books containing material on scaling and non-dimensionalization very often cover topics not treated in the present notes, e.g., the key topic of dimensional analysis and the famous Buckingham Pi Theorem [1, 9], which we discuss only briefly in section 1.1.3. Similarly, analytical solution methods like perturbation techniques and similarity solutions, which represent classical methods closely related to scaling and non-dimensionalization, are not addressed herein. There are numerous texts on perturbation techniques, and these methods build on an already scaled differential equations. Similarity solutions do not fit within the present scope since these involve non-dimensional *combinations* of the unscaled independent variables to derive new differential equations that are easier to solve.

Our scope is to scale differential equations to simplify the setting of parameters in numerical simulations, and at the same time understand more of the physics through interpretation of the dimensionless numbers that automatically arise from the scaling procedure.

With these notes, we hope to demystify the thinking involved in scale determination and encourage numerical simulations to be performed with dimensionless differential equation models.

All program and data files referred to in this book are available from the book's primary web site: URL: `http://hplgit.github.io/scaling-book/doc/web/`.

Acknowledgments. Professor Svein Linge provided very detailed, constructive comments on the entire manuscript and helped improve the reading quality significantly. Yapi Donatien Achou assisted with proof reading. The authors are thankful to Dr. Joakim Sundnes who contributed a section on scaling of the bidomain model in electrophysiology in the extended version of this booklet, found in Section 4.7. Significant portions of the present text were written when the first author was fed with FOLFIRINOX (and thereby kept alive) by Linda Falch-Koslung, Dr. Olav Dajani, and the rest of the OUS team. There would simply be no booklet without their efforts. It is also a great pleasure to express our sincere thanks to the Springer and Simula team that handled the prompt editing and production of the text: Martin Peters, Ruth Allewelt, Aslak Tveito, and Åsmund Ødegård.

Oslo, November 2015 *Hans Petter Langtangen, Geir K. Pedersen*

Contents

Preface v

1 Dimensions and units 1
- 1.1 Fundamental concepts 1
  - 1.1.1 Base units and dimensions 1
  - 1.1.2 Dimensions of common physical quantities 2
  - 1.1.3 The Buckingham Pi theorem 3
  - 1.1.4 Absolute errors, relative errors, and units 5
  - 1.1.5 Units and computers 5
  - 1.1.6 Unit systems 5
  - 1.1.7 Example on challenges arising from unit systems 6
  - 1.1.8 PhysicalQuantity: a tool for computing with units 7
- 1.2 Parampool: user interfaces with automatic unit conversion 9
  - 1.2.1 Pool of parameters 9
  - 1.2.2 Fetching pool data for computing 10
  - 1.2.3 Reading command-line options 11
  - 1.2.4 Setting default values in a file 12
  - 1.2.5 Specifying multiple values of input parameters 12
  - 1.2.6 Generating a graphical user interface 14

2 Ordinary differential equation models 17
- 2.1 Exponential decay problems 17
  - 2.1.1 Fundamental ideas of scaling 17
  - 2.1.2 The basic model problem 18
  - 2.1.3 The technical steps of the scaling procedure 19
  - 2.1.4 Making software for utilizing the scaled model 21
  - 2.1.5 Scaling a generalized problem 25
  - 2.1.6 Variable coefficients 30
  - 2.1.7 Scaling a cooling problem with constant temperature in the surroundings 32

2.1.8 Scaling a cooling problem with time-dependent surroundings ... 33
2.1.9 Scaling a nonlinear ODE ... 38
2.1.10 SIR ODE system for spreading of diseases ... 39
2.1.11 SIRV model with finite immunity ... 42
2.1.12 Michaelis-Menten kinetics for biochemical reactions ... 43
2.2 Vibration problems ... 49
2.2.1 Undamped vibrations without forcing ... 49
2.2.2 Undamped vibrations with constant forcing ... 53
2.2.3 Undamped vibrations with time-dependent forcing ... 54
2.2.4 Damped vibrations with forcing ... 61
2.2.5 Oscillating electric circuits ... 68
2.3 Exercises ... 69

3 Basic partial differential equation models ... 109
3.1 The wave equation ... 109
3.1.1 Homogeneous Dirichlet conditions in 1D ... 109
3.1.2 Implementation of the scaled wave equation ... 111
3.1.3 Time-dependent Dirichlet condition ... 112
3.1.4 Velocity initial condition ... 115
3.1.5 Variable wave velocity and forcing ... 117
3.1.6 Damped wave equation ... 120
3.1.7 A three-dimensional wave equation problem ... 121
3.2 The diffusion equation ... 121
3.2.1 Homogeneous 1D diffusion equation ... 122
3.2.2 Generalized diffusion PDE ... 123
3.2.3 Jump boundary condition ... 125
3.2.4 Oscillating Dirichlet condition ... 126
3.3 Reaction-diffusion equations ... 129
3.3.1 Fisher's equation ... 129
3.3.2 Nonlinear reaction-diffusion PDE ... 131
3.4 The convection-diffusion equation ... 132
3.4.1 Convection-diffusion without a force term ... 132
3.4.2 Stationary PDE ... 135
3.4.3 Convection-diffusion with a source term ... 137
3.5 Exercises ... 138

4 Advanced partial differential equation models ... 163
4.1 The equations of linear elasticity ... 163
4.1.1 The general time-dependent elasticity problem ... 163
4.1.2 Dimensionless stress tensor ... 165
4.1.3 When can the acceleration term be neglected? ... 166
4.1.4 The stationary elasticity problem ... 168
4.1.5 Quasi-static thermo-elasticity ... 169
4.2 The Navier-Stokes equations ... 171

4.2.1 The momentum equation without body forces 171
4.2.2 Scaling of time for low Reynolds numbers 173
4.2.3 Shear stress as pressure scale 174
4.2.4 Gravity force and the Froude number 174
4.2.5 Oscillating boundary conditions and the Strouhal number 175
4.2.6 Cavitation and the Euler number 176
4.2.7 Free surface conditions and the Weber number 176
4.3 Thermal convection 177
4.3.1 Forced convection 178
4.3.2 Free convection 179
4.3.3 The Grashof, Prandtl, and Eckert numbers 182
4.3.4 Heat transfer at boundaries and the Nusselt and Biot numbers 184
4.4 Compressible gas dynamics 185
4.4.1 The Euler equations of gas dynamics 185
4.4.2 General isentropic flow 187
4.4.3 The acoustic approximation for sound waves 189
4.5 Water surface waves driven by gravity 191
4.5.1 The mathematical model 191
4.5.2 Scaling 192
4.5.3 Waves in deep water 193
4.5.4 Long waves in shallow water 193
4.6 Two-phase porous media flow 194
4.7 The bidomain model in electrophysiology 198
4.7.1 The mathematical model 198
4.7.2 Scaling 200
4.7.3 An alternative I_{ion} 203
4.8 Exercises 204

References 211

Index 213

Chapter 1
Dimensions and units

A mechanical system undergoing one-dimensional damped vibrations can be modeled by the equation

$$mu'' + bu' + ku = 0, \tag{1.1}$$

where m is the mass of the system, b is some damping coefficient, k is a spring constant, and $u(t)$ is the displacement of the system. This is an equation expressing the balance of three physical effects: mu'' (mass times acceleration), bu' (damping force), and ku (spring force). The different physical quantities, such as m, $u(t)$, b, and k, all have different *dimensions*, measured in different *units*, but mu'', bu', and ku must all have the same dimension, otherwise it would not make sense to add them.

1.1 Fundamental concepts

1.1.1 Base units and dimensions

Base units have the important property that all other units derive from them. In the SI system, there are seven such base units and corresponding physical quantities: meter (m) for length, kilogram (kg) for mass, second (s) for time, kelvin (K) for temperature, ampere (A) for electric current, candela (cd) for luminous intensity, and mole (mol) for the amount of substance.

We need some suitable mathematical notation to calculate with dimensions like length, mass, time, and so forth. The dimension of length is written as [L], the dimension of mass as [M], the dimension of time as [T], and the dimension of temperature as [Θ] (the dimensions of the other base units are simply omitted as we do not make much use of them in this text). The dimension of a *derived unit* like velocity, which is distance (length) divided by time, then becomes [LT^{-1}] in this notation. The dimension of force, another derived unit, is the same as the dimension of mass times acceleration, and hence the dimension of force is [MLT^{-2}].

Let us find the dimensions of the terms in (1.1). A displacement $u(t)$ has dimension [L]. The derivative $u'(t)$ is change of displacement, which has dimension [L], divided by a time interval, which has dimension [T], implying that the dimension of u' is [LT^{-1}]. This result coincides with the interpretation of u' as velocity and the fact that velocity is defined as distance ([L]) per time ([T]).

Looking at (1.1), and interpreting $u(t)$ as displacement, we realize that the term mu'' (mass times acceleration) has dimension [MLT^{-2}]. The term bu' must have the same dimension, and since u' has dimension [LT^{-1}], b must have dimension [MT^{-1}]. Finally, ku must also have dimension [MLT^{-2}], implying that k is a parameter with dimension [MT^{-2}].

The unit of a physical quantity follows from the dimension expression. For example, since velocity has dimension [LT^{-1}] and length is measured in m while time is measured in s, the unit for velocity becomes m/s. Similarly, force has dimension [MLT^{-2}] and unit kg m/s^2. The k parameter in (1.1) is measured in kg s^{-2}.

Dimension of derivatives

The easiest way to realize the dimension of a derivative, is to express the derivative as a finite difference. For a function $u(t)$ we have

$$\frac{du}{dt} \approx \frac{u(t+\Delta t) - u(t)}{\Delta t},$$

where Δt is a small time interval. If u denotes a velocity, its dimension is $[\text{LT}]^{-1}$, and $u(t+\Delta t) - u(t)$ gets the same dimension. The time interval has dimension [T], and consequently, the finite difference gets the dimension $[\text{LT}]^{-2}$. In general, the dimension of the derivative du/dt is the dimension of u divided by the dimension of t.

1.1.2 Dimensions of common physical quantities

Many derived quantities are measured in derived units that have their own name. Force is one example: Newton (N) is a derived unit for force, equal to kg m/s^2. Another derived unit is Pascal (Pa) for pressure and stress, i.e., force per area. The unit of Pa then equals N/m^2 or kg/ms^2. Below are more names for derived quantities, listed with their units.

Name	Symbol	Physical quantity	Unit
radian	rad	angle	1
hertz	Hz	frequency	s^{-1}
newton	N	force, weight	kg m/s^2
pascal	Pa	pressure, stress	N/m^2
joule	J	energy, work, heat	Nm
watt	W	power	J/s

Some common physical quantities and their dimensions are listed next.

Quantity	Relation	Unit	Dimension
stress	force/area	$\mathrm{N/m^2 = Pa}$	$[\mathrm{MT^{-2}L^{-1}}]$
pressure	force/area	$\mathrm{N/m^2 = Pa}$	$\mathrm{MT^{-2}L^{-1}}]$
density	mass/volume	$\mathrm{kg/m^3}$	$[\mathrm{ML^{-3}}]$
strain	displacement/length	1	[1]
Young's modulus	stress/strain	$\mathrm{N/m^2 = Pa}$	$[\mathrm{MT^{-2}L^{-1}}]$
Poisson's ratio	transverse strain/axial strain	1	[1]
Lame' parameters λ and μ	stress/strain	$\mathrm{N/m^2 = Pa}$	$[\mathrm{MT^{-2}L^{-1}}]$
moment (of a force)	distance $\times$ force	Nm	$[\mathrm{ML^2T^{-2}}]$
impulse	force $\times$ time	Ns	$[\mathrm{MLT^{-1}}]$
linear momentum	mass $\times$ velocity	kg m/s	$[\mathrm{MLT^{-1}}]$
angular momentum	distance $\times$ mass $\times$ velocity	$\mathrm{kg\ m^2/s}$	$[\mathrm{ML^2T^{-1}}]$
work	force $\times$ distance	$\mathrm{Nm = J}$	$[\mathrm{ML^2T^{-2}}]$
energy	work	$\mathrm{Nm = J}$	$[\mathrm{ML^2T^{-2}}]$
power	work/time	$\mathrm{Nm/s = W}$	$[\mathrm{ML^2T^{-3}}]$
heat	work	J	$[\mathrm{ML^2T^{-2}}]$
heat flux	heat rate/area	$\mathrm{Wm^{-2}}$	$[\mathrm{MT^{-3}}]$
temperature	base unit	K	$[\Theta]$
heat capacity	heat change/temperature change	J/K	$[\mathrm{ML^2T^{-2}}\Theta^{-1}]$
specific heat capacity	heat capacity/unit mass	$\mathrm{JK^{-1}kg^{-1}}$	$[\mathrm{L^2T^{-2}}\Theta^{-1}]$
thermal conductivity	heat flux/temperature gradient	$\mathrm{Wm^{-1}K^{-1}}$	$[\mathrm{MLT^{-3}}\Theta^{-1}]$
dynamic viscosity	shear stress/velocity gradient	$\mathrm{kgm^{-1}s^{-1}}$	$[\mathrm{ML^{-1}}T^{-1}]$
kinematic viscosity	dynamic viscosity/density	$\mathrm{m^2/s}$	$[\mathrm{L^2T^{-1}}]$
surface tension	energy/area	$\mathrm{J/m^2}$	$[\mathrm{MT^{-2}}]$

Prefixes for units. Units often have prefixes. For example, kilo (k) is a prefix for 1000, so kg is 1000 g. Similarly, GPa means giga pascal or 10^9 Pa.

1.1.3 The Buckingham Pi theorem

Almost all texts on scaling has a treatment of the famous Buckingham Pi theorem, which can be used to derive physical laws based on unit compatibility rather than the underlying physical mechanisms. This booklet has its focus on models where the physical mechanisms are already expressed through differential equations. Nevertheless, the Pi theorem has a remarkable position in the literature on scaling, and since we will occasionally make references to it, the theorem is briefly discussed below.

The theorem itself is simply stated in two parts. First, if a problem involves n physical parameters in which m independent unit-types (such as length, mass etc.) appear, then the parameters can be combined to exactly $n-m$ independent dimensionless numbers, referred to as Pi's. Second, any unit-free relation between the original n parameters can be transformed into a relation between the $n-m$ dimensionless numbers. Such relations may be identities or inequalities stating, for instance, whether or not a given effect is negligible. Moreover, the transformation of an equation set into dimensionless form corresponds to expressing the coefficients, as well as the free and dependent variables, in terms of Pi's.

As an example, think of a body moving at constant speed v. What is the distance s traveled in time t? The Pi theorem results in one dimensionless variable $\pi = vt/s$ and leads to the formula $s = Cvt$, where C is an undetermined constant. The result is very close to the well-known formula $s = vt$ arising from the differential equation $s' = v$ in physics, but with an extra constant.

At first glance the Pi theorem may appear as bordering on the trivial. However, it may produce remarkable progress for selected problems, such as turbulent jets, nuclear blasts, or similarity solutions, without the detailed knowledge of mathematical or physical models. Hence, to a novice in scaling it may stand out as something very profound, if not magical. Anyhow, as one moves on to more complex problems with many parameters, the use of the theorem yields comparatively less gain as the number of Pi's becomes large. Many Pi's may also be recombined in many ways. Thus, good physical insight, and/or information conveyed through an equation set, is required to pick the useful dimensionless numbers or the appropriate scaling of the said equation set. Sometimes scrutiny of the equations also reveals that some Pi's, obtained by applying the theorem, in fact may be removed from the problem. As a consequence, when modeling a complex physical problem, the real assessment of scaling and dimensionless numbers will anyhow be included in the analysis of the governing equations instead of being a separate issue left with the Pi theorem. In textbooks and articles alike, the discussion of scaling in the context of the equations are too often missing or presented in a half-hearted fashion. Hence, the authors' focus will be on this process, while we do not provide much in the way of examples on the Pi theorem. We do not allude that the Pi theorem is of little value. In a number of contexts, such as in experiments, it may provide valuable and even crucial guidance, but in this particular textbook we seek to tell the complementary story on scaling. Moreover, as will be shown in this booklet, the dimensionless numbers in a problem also arise, in a very natural way, from scaling the differential equations. Provided one has a model based on differential equations, there is actually no need for classical dimensional analysis.

1.1.4 Absolute errors, relative errors, and units

Mathematically, it does not matter what units we use for a physical quantity. However, when we deal with approximations and errors, units are important. Suppose we work with a geophysical problem where the length scale is typically measured in km and we have an approximation 12.5 km to the exact value 12.52 km. The error is then 0.02 km. Switching units to mm leads to an error of 20,000 mm. A program working in mm would report $2 \cdot 10^5$ as the error, while a program working in km would print 0.02. The absolute error is therefore sensitive to the choice of units. This fact motivates the use of *relative error*: (exact - approximate)/exact, since units then cancel. In the present example, one gets a relative error of $1.6 \cdot 10^{-3}$ regardless of whether the length is measured in km or mm.

Nevertheless, rather than relying solely on relative errors, it is in general better to scale the problem such that the quantities entering the computations are of unit size (or at least moderate) instead of being very large or very small. The techniques of these notes show how this can be done.

1.1.5 Units and computers

Traditional numerical computing involves numbers only and therefore requires dimensionless mathematical expressions. Usually, an implicit trivial scaling is used. One can, for example, just scale all length quantities by 1 m, all time quantities by 1 s, and all mass quantities by 1 kg, to obtain the dimensionless numbers needed for calculations. This is the most common approach, although it is very seldom explicitly stated.

Symbolic computing packages, such as Mathematica and Maple, allow computations with quantities that have dimension. This is also possible in popular computer languages used for numerical computing (Section 1.1.8 provides a specific example in Python).

1.1.6 Unit systems

Confusion arises quickly when some physical quantities are expressed in SI units while others are in US or British units. Density could, for instance, be given in unit of ounce per teaspoon (see Exercise 2.1 for how to safely convert to a standard unit like $\mathrm{kg\,m^{-3}}$). Although unit conversion tables are frequently met in school, errors in unit conversion probably rank highest among all errors committed by scientists and engineers (and when a unit conversion error makes an airplane's fuel run out, it is serious!). Having good software tools to assist in unit conversion is therefore paramount, motivating the treatment of this topic in Sections 1.1.8 and 1.2. Readers who are primarily interested in the mathematical scaling technique may safely skip this material and jump right to Section 2.1.

1.1.7 Example on challenges arising from unit systems

A slightly elaborated example on scaling in an actual science/engineering project may stimulate the reader's motivation. In its full extent, the study of *tsunamis* spans geophysics, geology, history, fluid dynamics, statistics, geodesy, engineering, and civil protection. This complexity reflects in a diversity of practices concerning the use of units, scales, and concepts. If we narrow the scope to modeling of tsunami propagation, the scaling aspect, at least, may seem simple as we are mainly concerned with length and time. Still, even here the non-uniformity concerning physical units is an encumbrance.

A minor issue is the occasional use of non-SI units such as inches, or in old charts, even fathoms. More important is the non-uniformity in the magnitude of the different variables, and the differences in the inherent horizontal and vertical scales in particular. Typically, surface elevations are in meters or smaller. For far-field deep water propagation, as well as small tsunamis (which are still of scientific interest) surface elevations are often given in cm or even mm. In the deep ocean, the characteristic depth is orders of magnitude larger than this, typically 5000 m. Propagation distances, on the other hand, are hundreds or thousands of kilometers. Often locations and computational grids are best described in geographical coordinates (longitude/latitude) which are related to SI units by 1 latitude minute being roughly one nautical mile (1852 m), and 1 longitude minute being this quantity times the cosine of the latitude. Wave periods of tsunamis mostly range from minutes to an hour, hopefully sufficiently short to be well separated from the half-daily period of the tides. Propagation times are typically hours or maybe the better part of a day when the Pacific Ocean is traversed.

The scientists, engineers, and bureaucrats in the tsunami community tend to be particular and non-conform concerning formats and units, as well as the type of data required. To accommodate these demands, a tsunami modeler must produce a diversity of data which are in units and formats which cannot be used internally in her models. On the other hand, she must also be prepared to accept the input data in diversified forms. Some data sets may be large, implying that unnecessary duplication, with different units or scaling, should be avoided. In addition, tsunami models are often bench-marked through comparison with experimental data. The lab scale is generally cm or m, at most, which implies that measured data are provided in different units (than used in real earth-scale events), or even in volts, with conversion information, as obtained from the measuring gauges.

All the unit particulars in various file formats is clearly a nuisance and give rise to a number of misconceptions and errors that may cause loss of precious time or efforts. To reduce such problems, developers of computational tools should combine a reasonable flexibility concerning units in input and output with a clear and consistent convention for scaling within the tools. In fact, this also applies to academic tools for in-house use.

The discussion above points to some best practices that these notes promotes. First, always compute with scaled differential equation models. This booklet tells you how to do that. Second, users of software often want to specify input data with dimension and get output data with dimension. The software should then apply tools like `PhysicalQuantity` (Section 1.1.8) or the more sophisticated Parampool package (Section 1.2) to allow input with explicit dimensions and convert the dimensions to the right types if necessary. It is trivial to apply these tools if the computational software is written in Python, but it is even straightforward if the software is written in compiled languages like Fortran, C, or C++. In the latter case one just makes an input reading module in Python that grabs data from a user interface and feeds them into the computational software, either through files or function calls (the relevant functions to be called must be wrapped in Python with tools like f2py, Cython, Weave, SWIG, Instant, or similar, see [7, Appendix C] for basic examples on f2py and Cython wrapping of C and Fortran code).

1.1.8 PhysicalQuantity: a tool for computing with units

These notes contain quite some computer code to illustrate how the theory maps in detail to running software. Python is the programming language used, primarily because it is an easy-to-read, powerful, full-fledged language that allows MATLAB-like code as well as class-based code typically used in Java, C#, and C++. The Python ecosystem for scientific computing has in recent years grown fast in popularity and acts as a replacement for more specialized tools like MATLAB, R, and IDL. The coding examples in this booklet requires only familiarity with basic procedural programming in Python.

Readers without knowledge of Python variables, functions, if tests, and module import should consult, e.g., a brief tutorial on scientific Python, the Python Scientific Lecture Notes, or a full textbook [4] in parallel with reading about Python code in the present notes.

These notes apply Python 2.7

Python exists in two incompatible versions, numbered 2 and 3. The differences can be made small, and there are tools to write code that runs under both versions.

As Python version 2 is still dominating in scientific computing, we stick to this version, but write code in version 2.7 that is as close as possible to version 3.4 and later. In most of our programs, only the `print` statement differs between version 2 and 3.

Computations with units in Python are well supported by the very useful tool `PhysicalQuantity` from the ScientificPython package by Konrad Hinsen. Unfortunately, ScientificPython does not, at the time of this writing, work with NumPy version 1.9 or later, so we have isolated the `PhysicalQuantity` object in a module `PhysicalQuantities` and made it publicly available on GitHub. There is also an alternative package Unum for computing with numbers with units, but we shall stick to the former module here.

Let us demonstrate the usage of the `PhysicalQuantity` object by computing $s = vt$, where v is a velocity given in the unit *yards per minute* and t is time measured in hours. First we need to know what the units are called in `PhysicalQuantities`. To this end, run `pydoc PhysicalQuantities`, or

Terminal

```
Terminal> pydoc Scientific.Physics.PhysicalQuantities
```

if you have the entire ScientificPython package installed. The resulting documentation shows the names of the units. In particular, yards are specified by `yd`, minutes by `min`, and hours by `h`. We can now compute $s = vt$ as follows:

```
>>> # With ScientificPython:
>>> from Scientific.Physics.PhysicalQuantities import \
... PhysicalQuantity as PQ
>>> # With PhysicalQuantities as separate/stand-alone module:
>>> from PhysicalQuantities import PhysicalQuantity as PQ
>>>
>>> v = PQ('120 yd/min')   # velocity
>>> t = PQ('1 h')          # time
>>> s = v*t                # distance
>>> print s                # s is string
120.0 h*yd/min
```

The odd unit `h*yd/min` is better converted to a standard SI unit such as meter:

```
>>> s.convertToUnit('m')
>>> print s
6583.68 m
```

Note that `s` is a `PhysicalQuantity` object with a value and a unit. For mathematical computations we need to extract the value as a `float` object. We can also extract the unit as a string:

```
>>> print s.getValue()     # float
```

```
6583.68
>>> print s.getUnitName()  # string
m
```

Here is an example on how to convert the odd velocity unit yards per minute to something more standard:

```
>>> v.convertToUnit('km/h')
>>> print v
6.58368 km/h
>>> v.convertToUnit('m/s')
>>> print v
1.8288 m/s
```

As another example on unit conversion, say you look up the specific heat capacity of water to be 1 $\mathrm{cal\,g^{-1}K^{-1}}$. What is the corresponding value in the standard unit $\mathrm{Jg^{-1}K^{-1}}$ where joule replaces calorie?

```
>>> c = PQ('1 cal/(g*K)')
>>> c.convertToUnit('J/(g*K)')
>>> print c
4.184 J/K/g
```

1.2 Parampool: user interfaces with automatic unit conversion

The Parampool package allows creation of user interfaces with support for units and unit conversion. Values of parameters can be set as a number with a unit. The parameters can be registered beforehand with a preferred unit, and whatever the user prescribes, the value and unit are converted so the unit becomes the registered unit. Parampool supports various type of user interfaces: command-line arguments (option-value pairs), text files, and interactive web pages. All of these are described next.

Example application. As case, we want to make software for computing with the simple formula $s = v_0 t + \frac{1}{2}at^2$. We want v_0 to be a velocity with unit m/s, a to be acceleration with unit m/s^2, t to be time measured in s, and consequently s will be a distance measured in m.

1.2.1 Pool of parameters

First, Parampool requires us to define a *pool* of all input parameters, which is here simply represented by list of dictionaries, where each dictionary holds information about one parameter. It is possible to organize input parameters in a tree structure with subpools that themselves may have subpools, but for our simple application we just need a flat structure with three input parameters: v_0, a, and t. These parameters are put in a subpool called "Main". The pool is created by the code

```
def define_input():
    pool = [
        'Main', [
            dict(name='initial velocity', default=1.0, unit='m/s'),
            dict(name='acceleration', default=1.0, unit='m/s**2'),
            dict(name='time', default=10.0, unit='s')
            ]
        ]

    from parampool.pool.UI import listtree2Pool
    pool = listtree2Pool(pool)  # convert list to Pool object
    return pool
```

For each parameter we can define a logical name, such as `initial velocity`, a default value, and a unit. Additional properties are also allowed, see the Parampool documentation.

Tip: specify default values of numbers as float objects

Note that we do not just write 1, but 1.0 as default. Had 1 been used, Parampool would have interpreted our parameter as an integer and would therefore convert input like `2.5 m/s` to `2 m/s`. To ensure that a real-valued parameter becomes a `float` object inside the pool, we must specify the default value as a real number: `1.` or `1.0`. (The type of an input parameter can alternatively be set *explicitly* by the `str2type` property, e.g., `str2type=float`.)

1.2.2 Fetching pool data for computing

We can make a little function for fetching values from the pool and computing s:

```
def distance(pool):
    v_0 = pool.get_value('initial velocity')
    a = pool.get_value('acceleration')
    t = pool.get_value('time')
    s = v_0*t + 0.5*a*t**2
    return s
```

The `pool.get_value` function returns the numerical value of the named parameter, after the unit has been converted from what the user has specified to what was registered in the pool. For example, if the user provides the command-line argument `-time '2 h'`, Parampool will convert this quantity to seconds and `pool.get_value('time')` will return 7200.

1.2.3 Reading command-line options

To run the computations, we define the pool, load values from the command line, and call `distance`:

```
pool = define_input()
from parampool.menu.UI import set_values_from_command_line
pool = set_values_from_command_line(pool)

s = distance(pool)
print 's=%g' % s
```

Parameter names with whitespace must use an underscore for whitespace in the command-line option, such as in `--Initial_velocity`. We can now run

Terminal

```
Terminal> python distance.py --initial_velocity '10 km/h' \
          --acceleration 0 --time '1 h
s=10000
```

Notice from the answer (`s`) that 10 km/h gets converted to m/s and 1 h to s.

It is also possible to fetch parameter values as `PhysicalQuantity` objects from the pool by calling

```
v_0 = pool.get_value_unit('Initial velocity')
```

The following variant of the `distance` function computes with values and units:

```
def distance_unit(pool):
    # Compute with units
    from parampool.PhysicalQuantities import PhysicalQuantity as PQ
    v_0 = pool.get_value_unit('initial velocity')
    a = pool.get_value_unit('acceleration')
    t = pool.get_value_unit('time')
    s = v_0*t + 0.5*a*t**2
    return s.getValue(), s.getUnitName()
```

We can then do

```
s, s_unit = distance_unit(pool)
print 's=%g' % s, s_unit
```

and get output with the right unit as well.

1.2.4 Setting default values in a file

In large applications with lots of input parameters one will often like to define a (huge) set of default values specific for a case and then override a few of them on the command-line. Such sets of default values can be set in a file using syntax like

```
subpool Main
initial velocity = 100 ! yd/min
acceleration = 0 ! m/s**2         # drop acceleration
end
```

The unit can be given after the `!` symbol (and before the comment symbol `#`).

To read such files we have to add the lines

```
from parampool.pool.UI import set_defaults_from_file
pool = set_defaults_from_file(pool)
```

before the call to `set_defaults_from_command_line`.

If the above commands are stored in a file `distance.dat`, we give this file information to the program through the option `-poolfile distance.dat`. Running just

Terminal

```
Terminal> python distance.py --poolfile distance.dat
s=15.25 m
```

first loads the velocity 100 yd/min converted to 1.524 m/s and zero acceleration into the pool system and, and then we call `distance_unit`, which loads these values from the pool along with the default value for time, set as 10 s. The calculation is then $s = 1.524 \cdot 10 + 0 = 15.24$ with unit m. We can override the time and/or the other two parameters on the command line:

Terminal

```
Terminal> python distance.py --poolfile distance.dat --time '2 h'
s=10972.8 m
```

The resulting calculations are $s = 1.524 \cdot 7200 + 0 = 10972.8$. You are encouraged to play around with the `distance.py` program.

1.2.5 Specifying multiple values of input parameters

Parampool has an interesting feature: multiple values can be assigned to an input parameter, thereby making it easy for an application to run through all combinations of all parameters. We can demonstrate this feature by making a table of v_0, a, t, and s values. In the compute function, we need to call pool.get_values instead of pool.get_value to get a list of all the values that were specified for the parameter in question. By nesting loops over all parameters, we visit all combinations of all parameters as specified by the user:

```
def distance_table(pool):
    """Grab multiple values of parameters from the pool."""
    table = []
    for v_0 in pool.get_values('initial velocity'):
        for a in pool.get_values('acceleration'):
            for t in pool.get_values('time'):
                s = v_0*t + 0.5*a*t**2
                table.append((v_0, a, t, s))
    return table
```

In case just a single value was specified for a parameter, pool.get_values returns this value only and there will be only one pass in the associated loop.

After loading command-line arguments into our pool object, we can call distance_table instead of distance or distance_unit and write out a nicely formatted table of results:

```
table = distance_table(pool)
print '|-----------------------------------------------------|'
print '|      v_0    |      a     |       t     |      s       |'
print '|-----------------------------------------------------|'
for v_0, a, t, s in table:
    print '|%11.3f | %10.3f | %10.3f | %12.3f |' % (v_0, a, t, s)
print '|-----------------------------------------------------|'
```

Here is a sample run,

```
Terminal> python distance.py --time '1 h & 2 h & 3 h' \
          --acceleration '0 m/s**2 & 1 m/s**2 & 1 yd/s**2' \
          --initial_velocity '1 & 5'
|-----------------------------------------------------|
|      v_0    |      a     |       t     |      s       |
|-----------------------------------------------------|
|      1.000 |      0.000 |   3600.000 |     3600.000 |
|      1.000 |      0.000 |   7200.000 |     7200.000 |
|      1.000 |      0.000 |  10800.000 |    10800.000 |
|      1.000 |      1.000 |   3600.000 |  6483600.000 |
|      1.000 |      1.000 |   7200.000 | 25927200.000 |
|      1.000 |      1.000 |  10800.000 | 58330800.000 |
|      1.000 |      0.914 |   3600.000 |  5928912.000 |
|      1.000 |      0.914 |   7200.000 | 23708448.000 |
|      1.000 |      0.914 |  10800.000 | 53338608.000 |
|      5.000 |      0.000 |   3600.000 |    18000.000 |
|      5.000 |      0.000 |   7200.000 |    36000.000 |
|      5.000 |      0.000 |  10800.000 |    54000.000 |
|      5.000 |      1.000 |   3600.000 |  6498000.000 |
|      5.000 |      1.000 |   7200.000 | 25956000.000 |
|      5.000 |      1.000 |  10800.000 | 58374000.000 |
|      5.000 |      0.914 |   3600.000 |  5943312.000 |
|      5.000 |      0.914 |   7200.000 | 23737248.000 |
|      5.000 |      0.914 |  10800.000 | 53381808.000 |
```

```
|-------------------------------------------------------|
```

Notice that some of the multiple values have dimensions different from the registered dimension for that parameter, and the table shows that conversion to the right dimension has taken place.

1.2.6 Generating a graphical user interface

For the fun of it, we can easily generate a graphical user interface via Parampool. We wrap the `distance_unit` function in a function that returns the result in some nice-looking HTML code:

```
def distance_unit2(pool):
    # Wrap result from distance_unit in HTML
    s, s_unit = distance_unit(pool)
    return '<b>Distance:</b> %.2f %s' % (s, s_unit)
```

In addition, we must make a file `generate_distance_GUI.py` with the simple content

```
from parampool.generator.flask import generate
from distance import distance_unit2, define_input

generate(distance_unit2, pool_function=define_input, MathJax=True)
```

Running `generate_distance_GUI.py` creates a Flask-based web interface[1] to our `distance_unit` function, see Figure 1.1. The text fields in this GUI allow specification of parameters with numbers and units, e.g., acceleration with unit yards per minute squared, as shown in the figure. Hovering the mouse slightly to the left of the text field causes a little black window to pop up with the registered unit of that parameter.

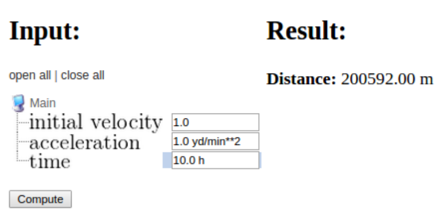


Fig. 1.1 Web GUI where parameters can be specified with units.

[1] You need to have Flask and additional packages installed. This is easy to do with a few `pip install` commands, see [5] or [6].

With examples shown above, the reader should be able to make use of the `PhysicalQuantity` object and the Parampool package in programs and thereby work safely with units. For the coming text, where we discuss the craft of scaling in detail, we shall just work in standard SI units and avoid unit conversion so there will be no more use of `PhysicalQuantity` and Parampool.

Chapter 2
Ordinary differential equation models

This chapter introduces the basic techniques of scaling and the ways to reason about scales. The first class of examples targets exponential decay models, starting with the simple ordinary differential equation (ODE) for exponential decay processes: $u' = -au$, with constant $a > 0$. Then we progress to various generalizations of this ODE, including nonlinear versions and systems of ODEs. The next class of examples concerns second-order ODEs for oscillatory systems, where the simplest ODE reads $mu'' + ku = 0$, with m and k as positive constants. Various extensions with damping and force terms are discussed in detail.

2.1 Exponential decay problems

2.1.1 Fundamental ideas of scaling

Scaling is an extremely useful technique in mathematical modeling and numerical simulation. The purpose of the technique is three-fold:

1. Make independent and dependent variables dimensionless.
2. Make the size of independent and dependent variables about unity.
3. Reduce the number of independent physical parameters in the model.

The first two items mean that for any variable, denote it by q, we introduce a corresponding dimensionless variable

$$\bar{q} = \frac{q - q_0}{q_c},$$

where q_0 is a reference value of q ($q_0 = 0$ is a common choice) and q_c is a characteristic size of $|q|$, often referred to as "a scale". Since the numerator and denominator have the same dimension, $\bar{q}$ becomes a dimensionless number.

If q_c is the maximum value of $|q - q_0|$, we see that $0 < |\bar{q}| \leq 1$. How to find q_c is sometimes the big challenge of scaling. Examples will illustrate various approaches to meet this challenge.

The many coming examples on scaling differential equations contain the following pedagogical ingredients to meet the desired learning outcomes.

- Teach the technical steps of making a mathematical model, based on differential equations, dimensionless.
- Describe various techniques for reasoning about the scales, i.e., finding the characteristic sizes of quantities.
- Teach how to identify and interpret dimensionless numbers arising from the scaling process.
- Provide a lot of different examples on making models dimensionless with physically correct scales.
- Show how symbolic software (SymPy) can be used to derive exact solutions of differential equations.
- Explain how to run a dimensionless model with software developed for the problem with dimensions.

2.1.2 The basic model problem

Processes undergoing exponential reduction can be modeled by the ODE problem

$$u'(t) = -au(t), \quad u(0) = I, \tag{2.1}$$

where $a, I > 0$ are prescribed parameters, and $u(t)$ is the unknown function. For the particular model with a constant a, we can easily derive the exact solution, $u(t) = Ie^{-at}$, which is helpful to have in mind during the scaling process.

Example: Population dynamics. The evolution of a population of humans, animals, cells, etc., under unlimited access to resources, can be modeled by (2.1). Then u is the number of individuals in the population, strictly speaking an integer, but well modeled by a real number in large populations. The parameter a is the increase in the number of individuals per time and per individual.

Example: Decay of pressure with altitude. The simple model (2.1) also governs the pressure in the atmosphere (under many assumptions, such air is an ideal gas in equilibrium). In this case u is the pressure, measured in Nm^{-2};

t is the height in meters; and $a = M/(R^*T)$, where M is the molar mass of the Earth's air (0.029 kg/mol), R^* is the universal gas constant ($8.314 \frac{\mathrm{Nm}}{\mathrm{mol\ K}}$), and T is the temperature in Kelvin (K). The temperature depends on the height so we have $a = a(t)$.

2.1.3 The technical steps of the scaling procedure

Step 1: Identify independent and dependent variables. There is one independent variable, t, and one dependent variable, u.

Step 2: Make independent and dependent variables dimensionless. We introduce a new dimensionless t, called $\bar{t}$, defined by

$$\bar{t} = \frac{t}{t_c}, \tag{2.2}$$

where t_c is a *characteristic value* of t. Similarly, we introduce a dimensionless u, named $\bar{u}$, according to

$$\bar{u} = \frac{u}{u_c}, \tag{2.3}$$

where u_c is a constant *characteristic size* of u. When u has a specific interpretation, say when (2.1) models pressure in an atmospheric layer, u_c would be referred to as characteristic pressure. For a decaying population, u_c may be a characteristic number of members in the population, e.g., the initial population I.

Step 3: Derive the model involving only dimensionless variables. The next task is to insert the new dimensionless variables in the governing mathematical model. That is, we replace t by $t_c\bar{t}$ and u by $u_c\bar{u}$ in (2.1). The derivative with respect to $\bar{t}$ is derived through the chain rule as

$$\frac{du}{dt} = \frac{d(u_c\bar{u})}{d\bar{t}}\frac{d\bar{t}}{dt} = u_c\frac{d\bar{u}}{d\bar{t}}\frac{1}{t_c} = \frac{u_c}{t_c}\frac{d\bar{u}}{d\bar{t}}\,.$$

The model (2.1) now becomes

$$\frac{u_c}{t_c}\frac{d\bar{u}}{d\bar{t}} = -au_c\bar{u}, \quad u_c\bar{u}(0) = I\,. \tag{2.4}$$

Step 4: Make each term dimensionless. Equation (2.4) still has terms with dimensions. To make each term dimensionless, we usually divide by the coefficient in front of the term with the highest time derivative (but dividing by any coefficient in any term will do). The result is

$$\frac{d\bar{u}}{d\bar{t}} = -at_c\bar{u}, \quad \bar{u}(0) = u_c^{-1}I\,. \tag{2.5}$$

Step 5: Estimate the scales. A characteristic quantity like t_c reflects the time scale in the problem. Estimating such a time scale is certainly the most challenging part of the scaling procedure. There are different ways to reason. The first approach is to aim at a size of $\bar{u}$ and its derivatives that is of order unity. If u_c is chosen such that $|\bar{u}|$ is of size unity, we see from (2.5) that $d\bar{u}/d\bar{t}$ is of the size of $\bar{u}$ (i.e., unity) if we choose $t_c = 1/a$.

Alternatively, we may look at a special case of the model where we have analytical insight that can guide the choice of scales. In the present problem we are lucky to know the exact solution for any value of the input data as long as a is a constant. For exponential decay, $u(t) \sim e^{-at}$, it is common to define a characteristic time scale t_c as the time it takes to reduce the initial value of u by a factor of $1/e$ (also called the *e-folding time*):

$$e^{-at_c} = \frac{1}{e}e^{-a\cdot 0} \quad \Rightarrow \quad e^{-at_c} = e^{-1},$$

from which it follows that $t_c = 1/a$. Note that using an exact solution of the problem to determine scales is not a requirement, just a useful help in the few cases where we actually have access to an exact solution.

In this example, two different, yet common ways of reasoning, lead to the same value of t_c. However, instead of using the e-folding time we could use the half-time of the exponential decay as characteristic time, which is also a very common measure of the time scale in such processes. The half time is defined as the time it takes to halve u:

$$e^{-at_c} = \frac{1}{2}e^{-a\cdot 0} \quad \Rightarrow \quad t_c = a^{-1}\ln 2\,.$$

There is a factor $\ln 2 = 0.69$ difference from the other t_c value. As long as the factor is not an order of magnitude or more different, we do not pay attention factors like $\ln 2$ and skip them, simply to make formulas look nicer. Using $t_c = a^{-1}\ln 2$ as time scale leads to a scaled differential equation $u' = -(\ln 2)u$, which is fine, but an unusual form. People tend to prefer the simpler ODE $u' = -u$, which arises from $t_c = 1/a$, and we shall therefore use this time scale.

Regarding u_c, we may look at the initial condition and realize that the choice $u_c = I$ makes $\bar{u}(0) = 1$. For $t > 0$, the differential equation expresses explicitly that u decreases, so $u_c = I$ gives $\bar{u} \in (0,1]$. Scaling a variable q such that $|\bar{q}| \in [0,1]$ is always the ultimate goal, and this goal is in fact obtained here! Next best result is to ensure that the magnitude of $|q|$ is not "big" or "small", in the sense that the size is neither as large as 10 or 100, nor as small as 0.1 or 0.01. (In the present problem, where we are lucky to have an exact solution $u(t) = Ie^{-at}$, we may look at this to explicitly see that $u \in (0,I]$ such that $u_c = I$ gives $\bar{u} \in (0,1]$).

With $t_c = 1/a$ and $u_c = I$, we have the final dimensionless model

$$\frac{d\bar{u}}{d\bar{t}} = -\bar{u}, \quad \bar{u}(0) = 1\,. \tag{2.6}$$

This is a remarkable result in the sense that *all physical parameters* (a and I) are removed from the model! Or more precisely, there are no physical input parameters to assign before using the model. In particular, numerical investigations of the original model (2.1) would need experiments with different a and I values, while numerical investigations of (2.6) can be limited to *a single run*! As soon as we have computed the curve $\bar{u}(\bar{t})$, we can find the solution $u(t)$ of (2.1) by

$$u(t) = u_c\bar{u}(t/t_c) = I\bar{u}(at)\,. \tag{2.7}$$

This particular transformation actually means stretching the $\bar{t}$ and $\bar{u}$ axes in a plot of $\bar{u}(\bar{t})$ by the factors a and I, respectively.

It is very common to drop the bars when the scaled problem has been derived and work further with (2.6) simply written as

$$\frac{du}{dt} = -u, \quad u(0) = 1\,.$$

Nevertheless, in this booklet we have decided to stick to bars for all dimensionless quantities.

2.1.4 Making software for utilizing the scaled model

Software for solving (2.1) could take advantage of the fact that only one simulation of (2.6) is necessary. As soon as we have $\bar{u}(\bar{t})$ accessible, a simple scaling (2.7) computes the real $u(t)$ for any given input data a and I. Although the numerical computation of $u(t)$ from (2.1) is very fast in this simple model problem, using (2.7) is very much faster. In general, a simple rescaling of a scaled solution is extremely more computationally efficient than solving a differential equation problem.

We can compute with the dimensionless model (2.6) in two ways, either make a solver for (2.6), or reuse a solver for (2.1) with $I=1$ and $a=1$. We will choose the latter approach since it has the advantage of giving us software that works both with a dimensionless model and a model with dimensions (and all the original physical parameters).

Software for the original unscaled problem. Assume that we have some module `decay.py` that offers the following functions:

- `solver(I, a, T, dt, theta=0.5)` for returning the solution arrays `u` and `t`, over a time interval $[0,T]$, for (2.1) solved by the so-called θ rule. This rule includes the Forward Euler scheme ($\theta=0$), the Backward Euler scheme ($\theta=1$), or the Crank-Nicolson (centered midpoint) scheme ($\theta=\frac{1}{2}$).
- `read_command_line_argparse()` for reading parameters in the problem from the command line and returning them: `I`, `a`, `T`, `theta` (θ), and a list of Δt values for time steps. (We shall only make use of the first Δt value.)

The basic statements for solving (2.1) are then

```
from decay import solver, read_command_line_argparse
I, a, T, theta, dt_values = read_command_line_argparse()
u, t = solver(I, a, T, dt_values[0], theta)

from matplotlib.pyplot import plot, show
plot(t, u)
show()
```

The module `decay.py` is developed and explained in Section 5.1.7 in [3].

To solve the dimensionless problem, just fix $I = 1$ and $a = 1$, and choose $\bar{T}$ and $\Delta\bar{t}$:

```
_, _, T, theta, dt_values = read_command_line_argparse()
u, t = solver(I=1, a=1, T=T, dt=dt_values[0], theta=theta)
```

The first two variables returned from `read_command_line_argparse` are `I` and `a`, which are ignored here. To indicate that these variables are not to be used, we use a "dummy name", often taken to be the underscore symbol in Python. The user can set `-I` and `-a` on the command line, since the `decay` module allows this, but we hope the code above has a form that reminds the user that these options are not to be used. Also note that `T` and `dt_values[0]` set on the command line are the desired parameters for solving the *scaled* problem.

Software for the scaled problem. Turning now to the scaled problem, the solver function (originally designed for the unscaled problem) will be reused, but it will only be run if it is strictly necessary. That is, when the user requests a solution, our code should first check whether that solution can be provided by simply scaling a solution already computed and available in a file. If not, we will compute an appropriate scaled solution, find the requested unscaled solution for the user, and also save the new scaled solution to file for possible later use.

A very plain solution to the problem is found in the file `decay_scaled_v1.py`. The `np.savetxt` function saves a two-dimensional array ("table") to a text file, and the `np.loadtxt` function can load the data back into the program. A better solution to this problem is obtained by using the `joblib` package as described next.

Implementation with joblib. The Python package `joblib` has functionality that is very convenient for implementing the `solver_scaled` function. The first time a function is called with a set of arguments, the statements in the function are executed and the return value is saved to file. If the function is called again with the same set of arguments, the statements in the function are not executed, but the return value is read from file (of course, many files may be stored, one for each combination of parameter values). In computer science, one would say that `joblib` in this way provides *memorization* functionality for Python functions. This functionality is particularly aimed at large-scale computations with arrays that one would hesitate to recompute. We illustrate the technique here in a very simple mathematical context.

First we make a `solver_scaled` function for the scaled model that just calls up a `solver_unscaled` (with $I = a = 1$) for the problem with dimensions:

```
from decay import solver as solver_unscaled
import numpy as np
import matplotlib.pyplot as plt

def solver_scaled(T, dt, theta):
    """
    Solve u'=-u, u(0)=1 for (0,T] with step dt and theta method.
    """
    print 'Computing the numerical solution'
    return solver_unscaled(I=1, a=1, T=T, dt=dt, theta=theta)
```

Then we create some "computer memory on disk", i.e., some disk space to store the result of a call to the `solver_scaled` function. Thereafter, we redefine the name `solver_scaled` to a new function, created by `joblib`, which calls our original `solver_scaled` function if necessary and otherwise loads data from file:

```
import joblib
disk_memory = joblib.Memory(cachedir='temp')
solver_scaled = disk_memory.cache(solver_scaled)
```

The solutions are actually stored in files in the cache directory `temp`.

A typical use case is to read values from the command line, solve the scaled problem (if necessary), unscale the solution, and visualize the solution with dimension:

```
def unscale(u_scaled, t_scaled, I, a):
    return I*u_scaled, a*t_scaled

from decay import read_command_line_argparse

def main():
    # Read unscaled parameters, solve and plot
    I, a, T, theta, dt_values = read_command_line_argparse()
    dt = dt_values[0]  # use only the first dt value
    T_bar = a*T
    dt_bar = a*dt
    u_scaled, t_scaled = solver_scaled(T_bar, dt_bar, theta)
    u, t = unscale(u_scaled, t_scaled, I, a)

    plt.figure()
    plt.plot(t_scaled, u_scaled)
    plt.xlabel('scaled time'); plt.ylabel('scaled velocity')
    plt.title('Universial solution of scaled problem')
    plt.savefig('tmp1.png');  plt.savefig('tmp1.pdf')

    plt.figure()
    plt.plot(t, u)
```

```
    plt.xlabel('t'); plt.ylabel('u')
    plt.title('I=%g, a=%g, theta=%g' % (I, a, theta))
    plt.savefig('tmp2.png'); plt.savefig('tmp2.pdf')
    plt.show()
```

The complete code resides in the file `decay_scaled.py`. Note from the code above that `read_command_line_argparse` is supposed to read parameters with dimensions (but technically, we solve the scaled problem, if strictly necessary, and unscale the solution). Let us run

Terminal

```
Terminal> python decay_scaled.py --I 8 --a 0.1 --dt 0.01 --T 50
```

A plot of the scaled and unscaled solution appears in Figure 2.1.

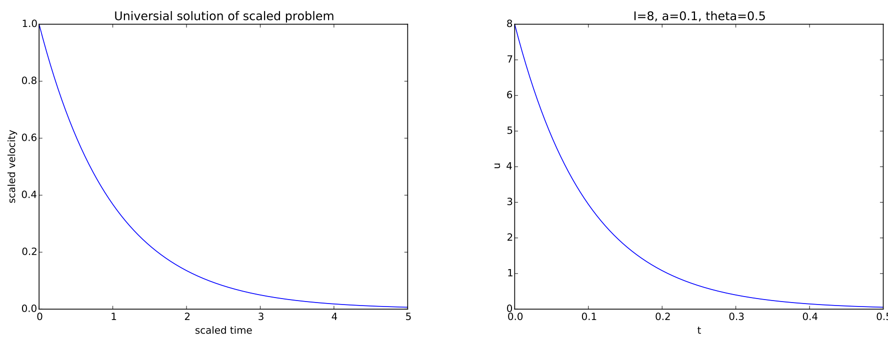


Fig. 2.1 Scaled (left) and unscaled (right) exponential decay.

Note that we write a message `Computing the numerical solution` inside the `solver_scaled` function. We can then easily detect when the solution is actually computed from scratch and when it is simply read from file (followed by the unscaling procedure). Here is a demo:

Terminal

```
Terminal> # Very first run
Terminal> python decay_scaled.py --T 7 --a 1 --I 0.5 --dt 0.2
[Memory] Calling __main__--home-hpl...
solver_scaled-alias(7.0, 0.2, 0.5)
Computing the numerical solution

Terminal> # No change of T, dt, theta - can reuse solution in file
Terminal> python decay_scaled.py --T 7 --a 4 --I 2.5 --dt 0.2

Terminal> # Change of dt, must recompute
Terminal> python decay_scaled.py --T 7 --a 4 --I 2.0 --dt 0.5
[Memory] Calling __main__--home-hpl...
solver_scaled-alias(7.0, 0.5, 0.5)
Computing the numerical solution
```

```
Terminal> # Change of dt again, but dt=0.2 is already in a file
Terminal> python decay_scaled.py --T 7 --a 0.5 --I 1 --dt 0.2
```

We realize that `joblib` has access to all previous runs and does not recompute unless it is strictly required. Our previous implementation without `joblib` (in `decay_scaled_v1.py`) used only one file (for one numerical case) and will therefore perform many more calls to `solver_unscaled`.

On the implementation of a simple memoize function

A memoized function recalls previous results when the same set of arguments is encountered. That is, the function caches its results. A simple implementation stores the arguments in a function call and the returned results in a dictionary, and if the arguments are seen again, one looks up in the dictionary and returns previously computed results:

```
class Memoize:
    def __init__(self, f):
        self.f = f
        self.memo = {}  # map arguments to results

def __call__(self, *args):
        if not args in self.memo:
            self.memo[args] = self.f(*args)
        return self.memo[args]

# Wrap my_compute_function(arg1, arg2, ...)
my_compute_function = Memoize(my_compute_function)
```

The memoize functionality in `joblib.Memory` is more sophisticated and can work very efficiently with large array data structures as arguments. Note that the simple version above can only be used when all arguments to the function `f` are immutable (since the key in a dictionary has to be immutable).

2.1.5 Scaling a generalized problem

Now we consider an extension of the exponential decay ODE to the form

$$u'(t) = -au(t) + b, \quad u(0) = I\,. \tag{2.8}$$

One particular model, with constant a and b, is a spherical small-sized organism falling in air,

$$u' = -\frac{3\pi d\mu}{\varrho_b V}u + g\left(\frac{\varrho}{\varrho_b} - 1\right), \tag{2.9}$$

where d, μ, ϱ_b, ϱ, V, and g are physical parameters. The function $u(t)$ represents the vertical velocity, being positive upwards. We shall use this model in the following.

Exact solution. It can be handy to have the exact solution for reference, in case of constant a and b:

$$u_{\mathrm{e}}(t) = \frac{e^{-at}}{a}\left(b(e^{at} - 1) + aI\right).$$

Solving differential equations in SymPy

It can be very useful to use a symbolic computation tool such as SymPy to aid us in solving differential equations. Let us therefore demonstrate how SymPy can be used to find this solution. First we define the parameters in the problem as symbols and $u(t)$ as a function:

```
>>> from sympy import *
>>> t, a, b, I = symbols('t a b I', real=True, positive=True)
>>> u = symbols('u', cls=Function)
```

The next task is to define the differential equation, either as a symbolic expression that is to equal zero, or as an equation `Eq(lhs, rhs)` with `lhs` and `rhs` as expressions for the left- and right-hand side):

```
>>> # Define differential equation
>>> eq = diff(u(t), t) + a*u(t) - b
>>> # or
>>> eq = Eq(diff(u(t), t), -a*u(t) + b)
```

The differential equation can be solved by the `dsolve` function, yielding an equation of the form `u(t) == expression`. We want to grab the expression on the right-hand side as our solution:

```
>>> sol = dsolve(eq, u(t))
>>> print sol
u(t) == (b + exp(a*(C1 - t)))/a
>>> u = sol.rhs                              # grab solution
>>> print u
(b + exp(a*(C1 - t)))/a
```

The solution contains the unknown integration constant `C1`, which must be determined by the initial condition. We form the equation arising from the initial condition $u(0) = I$:

```
>>> C1 = symbols('C1')
>>> eq = Eq(u.subs(t, 0), I)   # substitute t by 0 in u
```

```
>>> sol = solve(eq, C1)
>>> print sol
[log(I*a - b)/a]
```

The one solution that was found (stored in a list!) must then be substituted back in the expression `u` to yield the final solution:

```
>>> u = u.subs(C1, sol[0])
>>> print u
(b + exp(a*(-t + log(I*a - b)/a)))/a
```

As in mathematics with pen and paper, we strive to simplify expressions also in symbolic computing software. This frequently requires some trial and error process with SymPy's simplification functions. A very standard first try is to expand everything and run simplification algorithms:

```
>>> u = simplify(expand(u))
>>> print u
(I*a + b*exp(a*t) - b)*exp(-a*t)/a
```

Doing `latex(u)` automatically converts the expression to LaTeX syntax for inclusion in reports.

The reader may wonder why we bother with scaling of differential equations if SymPy can solved the problem in a nice, closed formula. This is true in the present introductory problem, but in a more general problem setting, we have some differential equation where SymPy perhaps can help with finding an exact solution only in a special case. We can use this special-case solution to control our reasoning about scales in the more general setting.

Theory. The challenges in our scaling is to find the right u_c and t_c scales. From (2.8) we see that if $u' \to 0$ as $t \to \infty$, u approaches the constant value b/a. It can be convenient to let the scaled $\bar{u} \to 1$ as we approach the $d\bar{u}/d\bar{t} = 0$ state. This idea points to choosing

$$u_c = \frac{b}{a} = g\left(\frac{\varrho}{\varrho_b} - 1\right)\left(\frac{3\pi d\mu}{\varrho_b V}\right)^{-1}. \tag{2.10}$$

On the sign of the scaled velocity

A little note on the sign of u_c is necessary here. With $\varrho_b < \varrho$, the buoyancy force upwards wins over the gravity force downwards, and the body will move upwards. In this case, the terminal velocity $u_c > 0$. When $\varrho_b > \varrho$, we get a motion downwards, and $u_c < 0$. The corresponding u is then also negative, but the scaled velocity u/u_c, becomes positive.

Inserting $u = u_c\bar{u} = b\bar{u}/a$ and $t = t_c\bar{t}$ in (2.8) leads to

$$\frac{d\bar{u}}{d\bar{t}} = -t_c a\bar{u} + \frac{t_c}{u_c} b, \quad \bar{u}(0) = I\frac{a}{b}\,.$$

We want the scales such that $d\bar{u}/d\bar{t}$ and $\bar{u}$ are about unity. To balance the size of $\bar{u}$ and $d\bar{u}/d\bar{t}$ we must therefore choose $t_c = 1/a$, resulting in the scaled ODE problem

$$\frac{d\bar{u}}{d\bar{t}} = -\bar{u} + 1, \quad \bar{u}(0) = \beta, \tag{2.11}$$

where β is a dimensionless number,

$$\beta = \frac{I}{u_c} = I\frac{a}{b}, \tag{2.12}$$

reflecting the ratio of the initial velocity and the terminal ($t \to \infty$) velocity b/a. Scaled equations normally end up with one or more dimensionless parameters, such as β here, containing ratios of physical effects in the model. Many more examples on dimensionless parameters will appear in later sections.

The analytical solution of the scaled model (2.11) reads

$$\bar{u_{\mathrm{e}}}(t) = e^{-t}\left(e^t - 1 + \beta\right) = 1 + (\beta - 1)e^{-t}\,. \tag{2.13}$$

The result (2.11) with the solution (2.13) is actually astonishing if a and b are as in (2.9): the six parameters d, μ, ϱ_b, ϱ, V, and g are conjured to one:

$$\beta = I\frac{3\pi d\mu}{\varrho_b V}\frac{1}{g}\left(\frac{\varrho}{\varrho_b} - 1\right)^{-1},$$

which is an enormous simplification of the problem if our aim is to investigate how u varies with the physical input parameters in the model. In particular, if the motion starts from rest, $\beta = 0$, and there are no physical parameters in the scaled model! We can then perform a single simulation and recover all physical cases by the unscaling procedure. More precisely, having computed $\bar{u}(\bar{t})$ from (2.11), we can use

$$u(t) = \frac{b}{a}\bar{u}(at), \tag{2.14}$$

to scale back to the original problem again. We observe that (2.11) can utilize a solver for (2.8) by setting $a = 1$, $b = 1$, and $I = \beta$. Given some implementation of a solver for (2.8), say `solver(I, a, b, T, dt, theta)`, the scaled model is run by `solver(beta, 1, 1, T, dt, theta)`.

Software. We may develop a solver for the scaled problem that uses `joblib` to cache solutions with the same β, Δt, and T. For now we fix $\theta = 0.5$. The module `decay_vc.py` (see Section 3.1.3 in [3] for details) has a function `solver(I, a, b, T, dt, theta)` for solving $u'(t) = -a(t)u(t) + b(t)$ for $t \in$

$(0,T]$, $u(0) = I$, with time step dt. We reuse this function and call it with $a = b = 1$ and $I = \beta$ to solve the scaled problem:

```
from decay_vc import solver as solver_unscaled

def solver_scaled(beta, T, dt, theta=0.5):
    """
    Solve u'=-u+1, u(0)=beta for (0,T]
    with step dt and theta method.
    """
    print 'Computing the numerical solution'
    return solver_unscaled(
        I=beta, a=lambda t: 1, b=lambda t: 1,
        T=T, dt=dt, theta=theta)

import joblib
disk_memory = joblib.Memory(cachedir='temp')
solver_scaled = disk_memory.cache(solver_scaled)
```

If we want to plot the physical solution, we need an `unscale` function,

```
def unscale(u_scaled, t_scaled, d, mu, rho, rho_b, V):
    a, b = ab(d, mu, rho, rho_b, V)
    return (b/a)*u_scaled, a*t_scaled

def ab(d, mu, rho, rho_b, V):
    g = 9.81
    a = 3*pi*d*mu/(rho_b*V)
    b = g*(rho/rho_b - 1)
    return a, b
```

Looking at droplets of water in air, we can fix some of the parameters and let the size parameter d be the one for experimentation. The following function sets physical parameters, computes β, runs the solver for the scaled problem (`joblib` detects if it is necessary), and finally plots the scaled curve $\bar{u}(\bar{t})$ and the unscaled curve $u(t)$.

```
def main(dt=0.075, # Time step, scaled problem
         T=7.5,    # Final time, scaled problem
         d=0.001,  # Diameter (unscaled problem)
         I=0,      # Initial velocity (unscaled problem)
         ):
    # Set parameters, solve and plot
    rho = 0.00129E+3  # air
    rho_b = 1E+3      # density of water
    mu = 0.001        # viscosity of water
    # Asumme we have list or similar for d
    if not isinstance(d, (list,tuple,np.ndarray)):
        d = [d]

    legends1 = []
    legends2 = []
    plt.figure(1)
    plt.figure(2)
```

```
    betas = []     # beta values already computed (for plot)

    for d_ in d:
        V = 4*pi/3*(d_/2.)**3  # volume
        a, b = ab(d_, mu, rho, rho_b, V)
        beta = I*a/b
        # Restrict to 3 digits in beta
        beta = abs(round(beta, 3))

        print 'beta=%.3f' % beta
        u_scaled, t_scaled = solver_scaled(beta, T, dt)

        # Avoid plotting curves with the same beta value
        if not beta in betas:
            plt.figure(1)
            plt.plot(t_scaled, u_scaled)
            plt.hold('on')
            legends1.append('beta=%g' % beta)
        betas.append(beta)

        plt.figure(2)
        u, t = unscale(u_scaled, t_scaled, d_, mu, rho, rho_b, V)
        plt.plot(t, u)
        plt.hold('on')
        legends2.append('d=%g [mm]' % (d_*1000))
    plt.figure(1)
    plt.xlabel('scaled time'); plt.ylabel('scaled velocity')
    plt.legend(legends1, loc='lower right')
```

The most complicated part of the code is related to plotting, but this part can be skipped when trying to understand how we work with a scaled model to perform the computations. The complete program is found in the file `falling_body.py`.

Since $I = 0$ implies $\beta = 0$, we can run different d values without any need to recompute $\bar{u}(\bar{t})$ as long as we assume the particle starts from rest.

From the scaling, we see that $u_c = b/a \sim d^{-2}$ and also that $t_c = 1/a \sim d^{-2}$, so plotting of $u(t)$ with dimensions for various d values will involve significant variations in the time and velocity scales. Figure 2.2 has an example with $d = 1, 2, 3$ mm, where we clearly see the different time and velocity scales in the figure with unscaled variables. Note that the scaled velocity is positive because of the sign of u_c (see the box above).

2.1.6 Variable coefficients

When a prescribed coefficient like $a(t)$ in $u'(t) = -a(t)u(t)$ varies with time one usually also performs a scaling of this a,

$$\bar{a}(\bar{t}) = \frac{a(t) - a_0}{a_c},$$

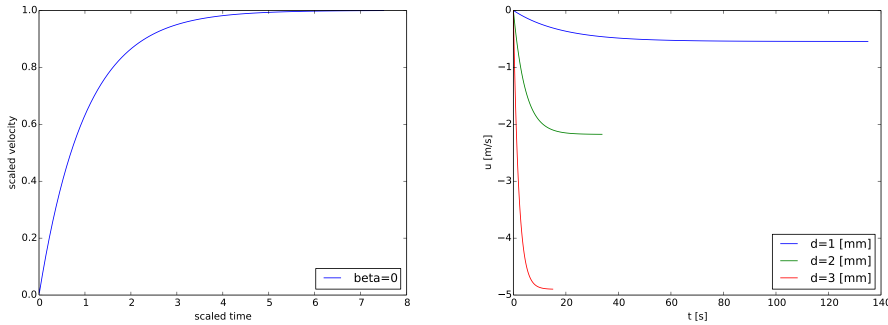


Fig. 2.2 Velocity of falling body: scaled (left) and with dimensions (right).

where the goal is to have the scaled $\bar{a}$ of size unity: $|\bar{a}| \leq 1$. This property is obtained by choosing a_c as the maximum value of $|a(t) - a_0|$ for $t \in [0, T]$, which is usually a quantity that can be estimated since $a(t)$ is known as a function of t. The a_0 parameter can be chosen as 0 here. (It could be tempting to choose $a_0 = \min_t a(t)$ so that $0 \leq \bar{a} \leq 1$, but then there is at least one point where $\bar{a} = 0$ and the differential equation collapses to $u' = 0$.)

As an example, imagine a decaying cell culture where we at time t_1 change the environment (typically the nutrition) such that the death rate increases by a factor 5. Mathematically, $a(t) = d$ for $t < t_1$ and $a(t) = 5d$ for $t \geq t_1$. The model reads $u' = -a(t)u$, $u(0) = I$.

The $a(t)$ function is scaled by letting the characteristic size be $a_c = d$ and $a_0 = 0$:

$$\bar{a}(\bar{t}) = \begin{cases} 1, \bar{t} < t_1/t_c \\ 5, \bar{t} \geq t_1/t_c \end{cases}$$

The scaled equation becomes

$$\frac{u_c}{t_c}\frac{d\bar{u}}{d\bar{t}} = a_c\bar{a}(\bar{t})u_c\bar{u}, \quad u_c\bar{u}(0) = I\,.$$

The natural choice of u_c is I. The characteristic time, previously taken as $t_c = 1/a$, can now be chosen as $t_c = t_1$ or $t_c = 1/d$. With $t_c = 1/d$ we get

$$\bar{u}'(\bar{t}) = -\bar{a}\bar{u}, \quad \bar{u}(0) = 1, \quad \bar{a} = \begin{cases} 1, \bar{t} < \gamma \\ 5, \bar{t} \geq \gamma \end{cases} \tag{2.15}$$

where

$$\gamma = t_1 d$$

is a dimensionless number in the problem. With $t_c = t_1$, we get

$$\bar{u}'(\bar{t}) = -\gamma\bar{a}\bar{u}, \quad \bar{u}(0) = 1, \quad \bar{a} = \begin{cases} 1, \bar{t} < 1 \\ 5, \bar{t} \geq 1 \end{cases}$$

The dimensionless parameter γ is now in the equation rather than in the definition of $\bar{a}$. Both problems involve γ, which is the ratio between the time when the environmental change happens and the typical time for the decay ($1/d$).

A computation with the scaled model (2.15) and the original model with dimensions appears in Figure 2.3.

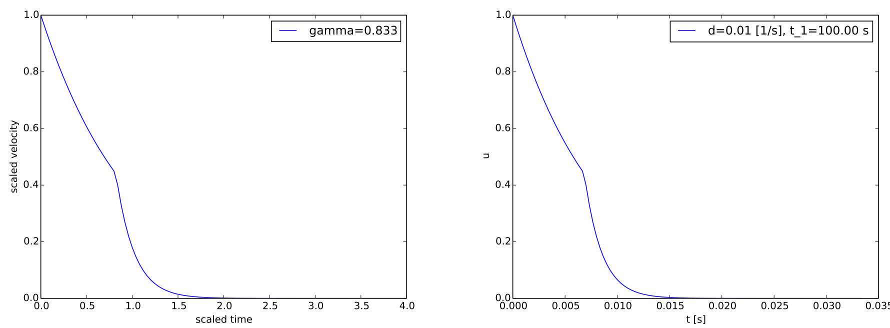


Fig. 2.3 Exponential decay with jump: scaled model (left) and unscaled model (right).

2.1.7 Scaling a cooling problem with constant temperature in the surroundings

The heat exchange between a body at temperature $T(t)$ and the surroundings at constant temperature T_s can be modeled by Newton's law of cooling:

$$T'(t) = -k(T - T_s), \quad T(0) = T_0, \tag{2.16}$$

where k is a prescribed heat transfer coefficient.

Exact solution. An analytical solution is always handy to have as a control of the choice of scales. The solution of (2.16) is by standard methods for ODEs found to be $T(t) = T_s + (T_0 - T_s)e^{-kt}$.

Scaling. Physically, we expect the temperature to start at T_0 and then to move toward the temperature of the surroundings (T_s). We therefore expect that T lies between T_0 and T_s. This is mathematically demonstrated by the analytical solution as well. A proper scaling is therefore to scale and translate T according to

$$\bar{T} = \frac{T - T_0}{T_s - T_0}\,. \tag{2.17}$$

Now, $0 \leq \bar{T} \leq 1$.

Scaling time by $\bar{t} = t/t_c$ and inserting $T = T_0 + (T_s - T_0)\bar{T}$ and $t = t_c\bar{t}$ in the problem (2.16) gives

$$\frac{d\bar{T}}{d\bar{t}} = -t_c k(\bar{T} - 1), \quad \bar{T}(0) = 0\,.$$

A natural choice, as argued in other exponential decay problems, is to choose $t_c k = 1$, which leaves us with the scaled problem

$$\frac{d\bar{T}}{d\bar{t}} = -(\bar{T} - 1), \quad \bar{T}(0) = 0\,. \tag{2.18}$$

No physical parameter enters this problem! Our scaling implies that $\bar{T}$ starts at 0 and approaches 1 as $\bar{t} \to \infty$, also in the case $T_s < T_0$. The physical temperature is always recovered as

$$T(t) = T_0 + (T_s - T_0)\bar{T}(k\bar{t})\,. \tag{2.19}$$

Software. An implementation for (2.16) works for (2.18) by setting $k = 1$, $T_s = 1$, and $T_0 = 0$.

Alternative scaling. An alternative temperature scaling is to choose

$$\bar{T} = \frac{T - T_s}{T_0 - T_s}\,. \tag{2.20}$$

Now $\bar{T} = 1$ initially and approaches zero as $t \to \infty$. The resulting scaled ODE problem then becomes

$$\frac{d\bar{T}}{d\bar{t}} = -\bar{T}, \quad \bar{T}(0) = 1,. \tag{2.21}$$

with solution $\bar{T} = e^{-\bar{t}}$.

2.1.8 Scaling a cooling problem with time-dependent surroundings

Let us apply the model (2.16) to the case when the surrounding temperature varies in time. Say we have an oscillating temperature environment according to

$$T_s(t) = T_m + a\sin(\omega t), \tag{2.22}$$

where T_m is the mean temperature in the surroundings, a is the amplitude of the variations around T_m, and $2\pi/\omega$ is the period of the temperature oscillations.

Exact solution. Also in this relatively simple problem it is possible to solve the differential equation problem analytically. Such a solution may be a good help to see what the scales are, and especially to control other forms for reasoning about the scales. Using the method of integrating factors for the original differential equation, we have

$$T(t) = T_0 e^{-kt} + e^{-kt} k \int_0^t e^{k\tau} T_s(\tau) d\tau \,.$$

With $T_s(t) = T_m + a\sin(\omega t)$ we can use SymPy to help us with integrations (note that we use `w` for ω in the computer code):

```
>>> from sympy import *
>>> t, k, T_m, a, w = symbols('t k T_m a w', real=True, positive=True)
>>> T_s = T_m + a*sin(w*t)
>>> I = exp(k*t)*T_s
>>> I = integrate(I, (t, 0, t))
>>> Q = k*exp(-k*t)*I
>>> Q = simplify(expand(Q))
>>> print Q
(-T_m*k**2 - T_m*w**2 + a*k*w +
(T_m*k**2 + T_m*w**2 + a*k**2*sin(t*w) -
a*k*w*cos(t*w))*exp(k*t))*exp(-k*t)/((k**2 + w**2))
```

Reordering the result, we get

$$T(t) = T_0 e^{-kt} + T_m(1-e^{-kt}) + (k^2+\omega^2)^{-1}(ak\omega e^{-kt} + ak\sin(\omega t) - akw\cos(\omega t))\,.$$

Scaling. The scaling (2.17) brings in a time-dependent characteristic temperature scale $T_s - T_0$. Let us start with a fixed scale, where we take the characteristic temperature variation to be $T_m - T_0$:

$$\bar{T} = \frac{T - T_0}{T_m - T_0}\,.$$

We realize by physical reasoning that T sets out at T_0, but with time, it will oscillate around T_m. (This reasoning can be controlled by looking at the exact solution we produced above.) The typical average temperature span is therefore $|T_m - T_0|$, unless a is much larger than $|T_m - T_0|$ or T_0 is very close to T_m (see Exercise 2.3 for a discussion of these cases).

We get from the differential equation, with $t_c = 1/k$ as in the former case,

$$k(T_m - T_0)\frac{d\bar{T}}{d\bar{t}} = -k((T_m - T_0)\bar{T} + T_0 - T_m - a\sin(\omega t),$$

resulting in

$$\frac{d\bar{T}}{d\bar{t}} = -\bar{T} + 1 + \alpha\sin(\beta\bar{t}), \quad \bar{T}(0) = 0, \tag{2.23}$$

where we have two dimensionless numbers:

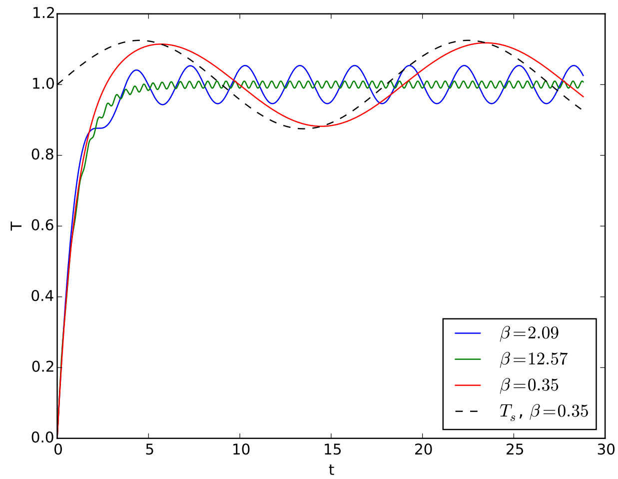
$$\alpha = \frac{a}{T_m - T_0}, \quad \beta = \frac{\omega}{k}\,.$$

The α quantity measures the ratio of temperatures: amplitude of oscillations versus distance from initial temperature to the average temperature for large times. The β number is the ratio of the two time scales: the frequency of the oscillations in T_s and the inverse e-folding time of the heat transfer. For clear interpretation of β we may introduce the period $P = 2\pi/\omega$ of the oscillations in T_s and the e-folding time $e = 1/k$. Then $\beta = 2\pi e/P$ and measures the e-folding time versus the period.

Remark

The original problem features five physical parameters: k, T_0, T_m, a, and ω, but only two dimensionless numbers appear in the scaled model (2.23). In fact, this is an example where application of the Pi theorem (see Section 1.1.3) falls short. Since, only time and temperature are involved as unit types, the theorem predicts that the five parameters yields three dimensionless numbers, not two. Scaling of the differential equations, on the other hand, shows us that the two parameters T_m and T_0 affect the nature of the problem only through their difference.

Software. Implementations of the unscaled problem (2.16) can be reused for the scaled model by setting $k = 1$, $T_0 = 0$, and $T_s(t) = 1 + \alpha\sin(\beta\bar{t})$ ($T_m = 1$, $a = \alpha$, $\omega = \beta$). The file `osc_cooling.py` contains solvers for the problem with dimensions and for the scaled problem. The figure below shows three cases of β values: small, medium, and large.



For the small β value, the oscillations in the surrounding temperature are slow enough compared to k for the heating and cooling process to follow the surrounding temperature, with a small time lag. For larger β, the heating and cooling require more time, and the oscillations get smaller.

Discussion of the time scale. There are two time variations of importance in the present problem: heat is transferred to the surroundings at a rate k, and the surroundings have a temperature variation with a period that goes like $1/\omega$. (We can, when we are so lucky that we have an analytical solution at hand, inspect this solution to see that k impacts the problem through a decay factor e^{-kt}, and ω impacts the problem through oscillations $\sin(\omega t)$.) The k parameter related to temperature decay points to a time scale $t_c = 1/k$, while the temperature oscillations of the surroundings point to a time scale $t_c = 1/\omega$. Which one should be chosen?

Bringing the temperature from T_0 to the level of the surroundings, T_m, goes like e^{-kt}, so in this process $t_c = 1/k$ is the characteristic time. Thereafter, the body's temperature just responds to the oscillations and the $\sin(\omega t)$ (and $\cos(\omega t)$) term dominates. For these large times, $t_c = 1/\omega$ is the appropriate time scale. Choosing $t_c = 1/\omega$ results in

$$\frac{d\bar{T}}{d\bar{t}} = -\beta^{-1}(\bar{T} - (1 + \alpha\sin(\bar{t}))), \quad \bar{T}(0) = 0\,. \tag{2.24}$$

Let us illustrate another, less effective, scaling. The temperature scale in (2.17) looks natural, so we apply this choice of scale. The characteristic temperature $T_0 - T_s$ now involves a time-dependent term $T_s(t)$. The mathematical steps become a bit more technically involved:

$$T(t) = T_0 + (T_s(t) - T_0)\bar{T},$$

$$\frac{dT}{dt} = \frac{dT_s}{dt}\bar{T} + (T_s - T_0)\frac{d\bar{T}}{d\bar{t}}\frac{d\bar{t}}{dt}\,.$$

With $\bar{t} = t/t_c = kt$ we get from the differential equation

$$\frac{dT_s}{dt}\bar{T} + (T_s - T_0)\frac{d\bar{T}}{d\bar{t}}k = -k(\bar{T} - 1)(T_s - T_0),$$

which after dividing by $k(T_s - T_0)$ results in

$$\frac{d\bar{T}}{d\bar{t}} = -(\bar{T} - 1) - \frac{dT_s}{dt}\frac{\bar{T}}{k(T_s - T_0},$$

or

$$\frac{d\bar{T}}{d\bar{t}} = -(\bar{T} - 1) - \frac{a\omega\cos(\omega\bar{t}/k)}{k(T_m + a\sin(\omega\bar{t}/k) - T_0)}\bar{T}\,.$$

The last term is complicated and becomes more tractable if we factor out dimensionless numbers. To this end, we scale T_s by (e.g.) T_m, which means to factor out T_m in the denominator. We are then left with

$$\frac{d\bar{T}}{d\bar{t}} = -(\bar{T} - 1) - \alpha\beta\frac{\cos(\beta\bar{t})}{1 + \alpha\sin(\beta\bar{t}) - \gamma}\bar{T}, \tag{2.25}$$

where α, β, and γ are dimensionless numbers characterizing the relative importance of parameters in the problem:

$$\alpha = a/T_m, \quad \beta = \omega/k, \quad \gamma = T_0/T_m\,. \tag{2.26}$$

We notice that (2.25) is not a special case of the original problem (2.16). Furthermore, the original five parameters k, T_m, a, ω, and T_0 are reduced to three dimensionless parameters. We conclude that this scaling is inferior, because using the temperature scale $T_0 - T_m$ enables reuse of the software for the unscaled problem and only two dimensionless parameters appear in the scaled model.

Let us briefly mention another possible temperature scaling: $\bar{T} = T/T_m$, motivated by the fact that as $t \to \infty$, T will oscillate around T_m, so this $\bar{T}$ will oscillate around unity. We get the dimensionless ODE

$$\frac{d\bar{T}}{d\bar{t}} = -(\bar{T} - (1 + \delta\sin(\beta\bar{t}))),$$

with a new dimensionless parameter $\delta = a/T_m$. However, the initial condition becomes $\bar{T}(0) = T_0/T_m$, and the ratio T_0/T_m is a third dimensionless parameter, so this scaling is also inferior to the one above with only two parameters.

2.1.9 Scaling a nonlinear ODE

Exponential growth models, $u' = au$, are not realistic in environments with limited resources. However, by letting a depend on u, the effect of limited resources can well be captured by such a simple differential equation model:

$$u' = a(u)u, \quad u(0) = I\,. \tag{2.27}$$

If the maximum value of u is denoted by M, we have that $a(M) = 0$. A simple choice fulfilling this requirement is $a(u) = \varrho(1-u/M)$. The parameter ϱ can be interpreted as the initial exponential growth rate if we assume that $I/M \ll 1$, since at $t = 0$ the model then approximates $u' = \varrho u$.

The choice $a(u) = \varrho(1-u/M)$ is known as the logistic model for population growth:

$$u' = \varrho u(1-u/M), \quad u(0) = I\,. \tag{2.28}$$

A more complicated choice of a may be $a(u) = \varrho(1-u/M)^p$ for some exponent p (this function also fulfills $a(M) = 0$ and $a \approx \varrho$ for $t = 0$).

Scaling. Let us scale (2.27) with $a(u) = \varrho(1-u/M)^p$. The natural scale for u is M ($u_c = M$), since we know that $0 < u \leq M$, and this makes the dimensionless $\bar{u} = u/M \in (0,1]$. The function $a(u)$ is typically varying between 0 and ϱ, so it can be scaled as

$$\bar{a}(\bar{u}) = \frac{a(u)}{\varrho} = (1 - \frac{u}{M})^p = (1-\bar{u})^p\,.$$

Time is scaled as $\bar{t} = t/t_c$ for some suitable characteristic time t_c. Inserted in (2.27), we get

$$\frac{u_c}{t_c}\frac{d\bar{u}}{d\bar{t}} = \varrho\bar{a}u_c\bar{u}, \quad u_c\bar{u}(0) = I,$$

resulting in

$$\frac{d\bar{u}}{d\bar{t}} = t_c\varrho(1-\bar{u})^p\bar{u}, \quad \bar{u}(0) = \frac{I}{M}\,.$$

A natural choice is $t_c = 1/\varrho$ as in other exponential growth models since it leads to the term on the right-hand side to be about unity, just as the left-hand side. (If the scaling is correct, $\bar{u}$ and its derivatives are of order unity, so the coefficients must also be of order unity.) Introducing also the dimensionless parameter

$$\alpha = \frac{I}{M},$$

measuring the fraction of the initial population compared to the maximum one, we get the dimensionless model

$$\frac{d\bar{u}}{d\bar{t}} = (1-\bar{u})^p \bar{u}, \quad \bar{u}(0) = \alpha\,. \tag{2.29}$$

Here, we have two dimensionless parameters: α and p. A classical logistic model with $p = 1$ has only one dimensionless variable.

Alternative scaling. We could try another scaling of u where we also translate $\bar{u}$:

$$\bar{u} = \frac{u - I}{M}\,.$$

This choice of $\bar{u}$ results in

$$\frac{d\bar{u}}{d\bar{t}} = (1-\alpha-\bar{u})^p \bar{u}, \quad \bar{u}(0) = 0\,. \tag{2.30}$$

The essential difference between (2.29) and (2.30) is that $\bar{u} \in [\alpha, 1]$ in the former and $\bar{u} \in [0, 1-\alpha]$ in the latter. Both models involve the dimensionless numbers α and p. An advantage of (2.29) is that software for the unscaled model can easily be used for the scaled model by choosing $I = \alpha$, $M = 1$, and $\varrho = 1$.

2.1.10 SIR ODE system for spreading of diseases

The field of epidemiology frequently applies ODE systems to describe the spreading of diseases, such as smallpox, measles, plague, ordinary flu, swine flu, and HIV. Different models include different effects, which are reflected in dimensionless numbers. Most of the effects are modeled as exponential decay or growth of the dependent variables.

The simplest model has three categories of people: susceptibles (S) who can get the disease, infectious (I) who are infected and may infect susceptibles, and recovered (R) who have recovered from the disease and gained immunity. We introduce $S(t)$, $I(t)$, and $R(t)$ as the number of people in the categories S, I, and R, respectively. The model, naturally known as the SIR model, can be expressed as a system of three ODEs:

$$\frac{dS}{dt} = -\beta S I, \tag{2.31}$$

$$\frac{dI}{dt} = \beta S I - \nu I, \tag{2.32}$$

$$\frac{dR}{dt} = \nu I, \tag{2.33}$$

where β and ν are empirical constants. The average time for recovering from the disease can be shown to be ν^{-1}, but β is much harder to estimate, so working with a scaled model where β is "scaled away" is advantageous.

Scaling. It is natural to scale S, I, and R by, e.g., $S(0)$:

$$\bar{S} = \frac{S}{S(0)}, \quad \bar{I} = \frac{I}{S(0)}, \quad \bar{R} = \frac{R}{S(0)}\,.$$

Introducing $\bar{t} = t/t_c$, we arrive at the equations

$$\begin{aligned}
\frac{d\bar{S}}{d\bar{t}} &= -t_c S(0)\beta\bar{S}\bar{I},\\
\frac{d\bar{I}}{d\bar{t}} &= t_c S(0)\beta\bar{S}\bar{I} - t_c\nu\bar{I},\\
\frac{d\bar{R}}{d\bar{t}} &= t_c\nu\bar{I},
\end{aligned}$$

with initial conditions $\bar{S}(0) = 1$, $\bar{I}(0) = I_0/S(0) = \alpha$, and $\bar{R}(0) = R(0)/S(0)$. Normally, $R(0) = 0$.

Taking $t_c = 1/\nu$, corresponding to a time unit equal to the time it takes to recover from the disease, we end up with the scaled model

$$\frac{d\bar{S}}{d\bar{t}} = -R_0\bar{S}\bar{I}, \tag{2.34}$$

$$\frac{d\bar{I}}{d\bar{t}} = R_0\bar{S}\bar{I} - \bar{I}, \tag{2.35}$$

$$\frac{d\bar{R}}{d\bar{t}} = \bar{I}, \tag{2.36}$$

with $\bar{S}(0) = 1$, $\bar{I}(0) = \alpha$, $\bar{R}(0) = 0$, and R_0 as the dimensionless number

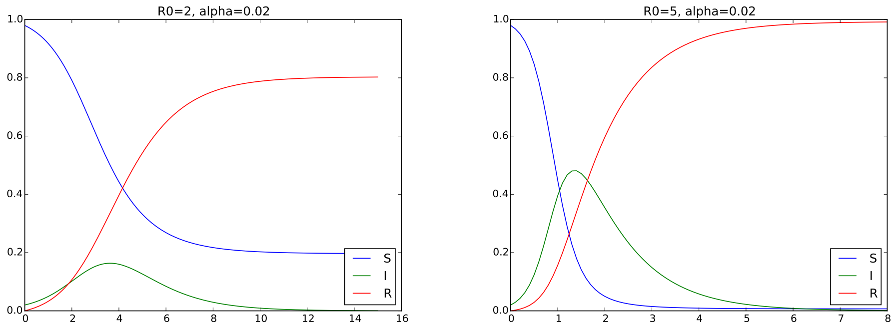
$$R_0 = \frac{S(0)\beta}{\nu}\,. \tag{2.37}$$

We see from (2.35) that to make the disease spreading, $d\bar{I}/d\bar{t} > 0$, and therefore $R_0\bar{S}(0) - 1 > 0$ or $R_0 > 1$ since $\bar{S}(0) = 1$. Therefore, R_0 reflects the disease's ability to spread and is consequently an important dimensionless quantity, known as the basic reproduction number. This number reflects the number of infected people caused by one infectious individual during the time period of the disease.

Looking at (2.32), we see that to increase I initially, we must have $dI/dt > 0$ at $t = 0$, which implies $\beta I(0)S(0) - \nu I(0) > 0$, i.e., $R_0 > 1$.

Software. Any implementation of the SIR model with dimensions can be reused for the scaled model by setting $\beta = R_0$, $\nu = 1$, $S(0) = 1 - \alpha$, and

$I(0) = \alpha$. Below is a plot with two cases: $R_0 = 2$ and $R_0 = 5$, both with $\alpha = 0.02$.



Alternative scaling. Adding (2.31)-(2.33) shows that

$$\frac{dS}{dt} + \frac{dI}{dt} + \frac{dR}{dt} = 0 \quad \Rightarrow \quad S + I + R = \text{const} = N,$$

where N is the size of the population. We can therefore scale S, I, and R by the total population $N = S(0) + I(0) + R(0)$:

$$\bar{S} = \frac{S}{N}, \quad \bar{I} = \frac{I}{N}, \quad \bar{R} = \frac{R}{N}.$$

With the same time scale, one gets the system (2.34)-(2.36), but with R_0 replaced by the dimensionless number:

$$\tilde{R}_0 = \frac{N\beta}{\nu}. \tag{2.38}$$

The initial conditions become $\bar{S}(0) = 1 - \alpha$, $\bar{I}(0) = \alpha$, and $\bar{R}(0) = 0$.

For the disease to spread at $t = 0$, we must have $\tilde{R}_0 \bar{S}(0) > 1$, but $\tilde{R}_0 \bar{S}(0) = N\beta/\nu \cdot S(0)/N = R_0$, so the criterion is still $R_0 > 1$. Since R_0 is a more famous number than $\tilde{R}_0$, we can write the ODEs with $R_0/S(0) = R_0/(1-\alpha)$ instead of $\tilde{R}_0$.

Choosing t_c to make the SI terms balance the time derivatives, $t_c = (N\beta)^{-1}$, moves $\tilde{R}_0$ (or R_0 if we scale S, I, and R by $S(0)$) to the I terms:

$$\begin{aligned}
\frac{d\bar{S}}{d\bar{t}} &= -\bar{S}\bar{I}, \\
\frac{d\bar{I}}{d\bar{t}} &= \bar{S}\bar{I} - \tilde{R}_0^{-1}\bar{I}, \\
\frac{d\bar{R}}{d\bar{t}} &= \tilde{R}_0^{-1} I.
\end{aligned}$$

2.1.11 SIRV model with finite immunity

A common extension of the SIR model involves finite immunity: after some period of time, recovered individuals lose their immunity and become susceptibles again. This is modeled as a leakage $-\mu R$ from the R to the S category, where μ^{-1} is the average time it takes to lose immunity. Vaccination is another extension: a fraction pS is removed from the S category by successful vaccination and brought to a new category V (the vaccinated). The ODE model reads

$$\frac{dS}{dt} = -\beta SI - pS + \mu R, \tag{2.39}$$
$$\frac{dI}{dt} = \beta SI - \nu I, \tag{2.40}$$
$$\frac{dR}{dt} = \nu I - \mu R, \tag{2.41}$$
$$\frac{dV}{dt} = pS\,. \tag{2.42}$$

Using $t_c = 1/\nu$ and scaling the unknowns by $S(0)$, we arrive at the dimensionless model

$$\frac{d\bar{S}}{d\bar{t}} = -R_0\bar{S}\bar{I} - \delta\bar{S} + \gamma\bar{R}, \tag{2.43}$$
$$\frac{d\bar{I}}{d\bar{t}} = R_0\bar{S}\bar{I} - \bar{I}, \tag{2.44}$$
$$\frac{d\bar{R}}{d\bar{t}} = \bar{I} - \gamma\bar{R}, \tag{2.45}$$
$$\frac{d\bar{V}}{d\bar{t}} = \delta\bar{S}, \tag{2.46}$$

with two new dimensionless parameters:

$$\gamma = \frac{\mu}{\nu}, \quad \delta = \frac{p}{\nu}\,.$$

The quantity p^{-1} can be interpreted as the average time it takes to vaccinate a susceptible successfully. Writing $\gamma = \nu^{-1}/\mu^{-1}$ and $\delta = \nu^{-1}/p^{-1}$ gives the interpretation that γ is the ratio of the average time to recover and the average time to lose immunity, while δ is the ratio of the average time to recover and the average time to successfully vaccinate a susceptible.

The plot in Figure 2.4 has $\gamma = 0.05$, i.e., loss of immunity takes 20 weeks if it takes one week to recover from the disease. The left plot corresponds to no vaccination, while the right has $\delta = 0.5$ for a vaccination campaign that lasts from day 7 to day 15. The value $\delta = 0.5$ reflects that it takes two weeks to successfully vaccinate a susceptible, but the effect of vaccination is still dramatic.

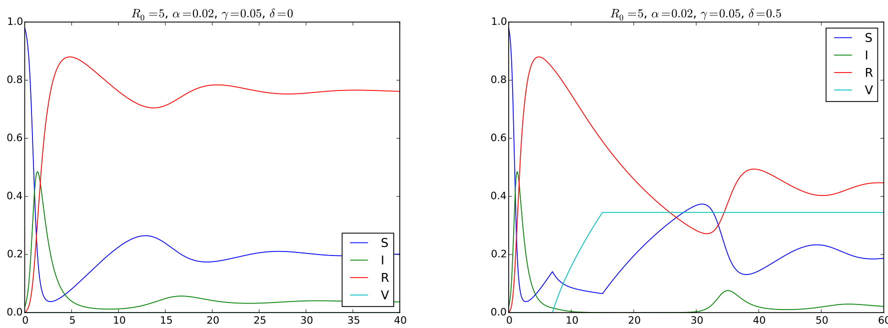


Fig. 2.4 Spreading of a disease with loss of immunity (left) and added vaccination (right).

2.1.12 Michaelis-Menten kinetics for biochemical reactions

A classical reaction model in biochemistry describes how a substrate S is turned into a product P with aid of an enzyme E. S and E react to form a complex ES in the first stage of the reaction. In the second stage, ES is turned into E and P. Introducing the amount of S, E, ES, and P by $[S]$, $[E]$, $[ES]$, and $[P]$, the mathematical model can be written as

$$\frac{d[ES]}{dt} = k_+[E][S] - k_v[ES] - k_-[ES], \tag{2.47}$$

$$\frac{d[P]}{dt} = k_v[ES], \tag{2.48}$$

$$\frac{d[S]}{dt} = -k_+[E][S] + k_-[ES], \tag{2.49}$$

$$\frac{d[E]}{dt} = -k_+[E][S] + k_-[ES] + k_v[ES]\,. \tag{2.50}$$

The initial conditions are $[ES](0) = [P](0) = 0$, and $[S] = S_0$, $[E] = E_0$. Three rate constants are involved: k_+, k_-, and k_v. The above mathematical model is known as Michaelis-Menten kinetics.

The amount of substance is measured in the unit mole (mol). From the equations we can see that k_+ is measured in $\mathrm{s}^{-1}\mathrm{mol}^{-1}$, while k_- and k_v are measured in s^{-1}. It is convenient to get rid of the mole unit for the amount of a substance. When working with dimensionless quantities, only ratios of the rate constants and not their specific values are needed.

Classical analysis. A common assumption is that the formation of $[ES]$ is very fast and that it quickly reaches an equilibrium state, $[ES]'=0$. Equation (2.47) then reduces to the algebraic equation

$$k_+[E][S]-k_v[ES]-k_-[ES]=0,$$

which leads to

$$\frac{[E][S]}{[ES]}=\frac{k_-+k_v}{k_+}=K, \tag{2.51}$$

where K is the famous Michaelis constant - the equilibrium constant between $[E][S]$ and $[ES]$.

Another important observation is that the ODE system implies two conservation equations, arising from simply adding the ODEs:

$$\frac{d[ES]}{dt}+\frac{d[E]}{dt}=0, \tag{2.52}$$
$$\frac{d[ES]}{dt}+\frac{d[S]}{dt}+\frac{d[P]}{dt}=0, \tag{2.53}$$

from which it follows that

$$[ES]+[E]=E_0, \tag{2.54}$$
$$[ES]+[S]+[P]=S_0\,. \tag{2.55}$$

We can use (2.54) and (2.51) to express $[E]$ by $[S]$:

$$[E]=E_0-[ES]=E_0-\frac{[E][S]}{K} \quad \Rightarrow \quad [E]=\frac{KE_0}{K+[S]}\,.$$

Now (2.49) can be developed to an equation involving $[S]$ only:

$$\begin{aligned}\frac{d[S]}{dt}&=-k_+[E][S]+k_-[ES]\\&=(-k_++\frac{k_-}{K})[E][S]\\&=(-k_++\frac{k_-}{K})[S]\frac{KE_0}{K+[S]}\\&=-\frac{k_-E_0}{[S]+K}\,.\end{aligned} \tag{2.56}$$

We see that the parameter K is central.

From above expression for $[E]$ and (2.54) it now follows

$$[E] = \frac{KE_0}{K+[S]}, \quad [ES] = \frac{E_0[S]}{K+[S]}.$$

If K is comparable to S_0 these indicate

$$[E] \sim E_0, \quad [ES] \sim \frac{E_0 S_0}{K},$$

as is used for scaling $[E]$ and Q_c, subsequently. Provided we exclude the case $[S] \gg K$, we may infer that $[E]$ will be of magnitude E_0, while $[ES]$ will be of magnitude $E_0 S_0/K$.

Dimensionless ODE system. Let us reason how to make the original ODE system (2.47)-(2.50) dimensionless. Aiming at $[S]$ and $[E]$ of unit size, two obvious dimensionless unknowns are

$$\bar{S} = \frac{[S]}{S_0}, \quad \bar{E} = \frac{[E]}{E_0}.$$

For the other two unknowns we just introduce scales to be determined later:

$$\bar{P} = \frac{[P]}{P_c}, \quad \bar{Q} = \frac{[ES]}{Q_c}.$$

With $\bar{t} = t/t_c$ the equations become

$$\begin{aligned}
\frac{d\bar{Q}}{d\bar{t}} &= t_c k_+ \frac{E_0 S_0}{Q_c} \bar{E}\bar{S} - t_c(k_v + k_-)\bar{Q}, \\
\frac{d\bar{P}}{d\bar{t}} &= t_c k_v \frac{Q_c}{P_c} \bar{Q}, \\
\frac{d\bar{S}}{d\bar{t}} &= -t_c k_+ E_0 \bar{E}\bar{S} + t_c k_- \frac{Q_c}{S_0} \bar{Q}, \\
\frac{d\bar{E}}{d\bar{t}} &= -t_c k_+ S_0 \bar{E}\bar{S} + t_c(k_- + k_v) \frac{Q_c}{E_0} \bar{Q}.
\end{aligned}$$

Determining scales. Choosing the scales is actually a quite complicated matter that requires extensive analysis of the equations to determine the characteristics of the solutions. Much literature is written about this, but here we shall take a simplistic and pragmatic approach. Besides the Michaelis constant K, there is another important parameter,

$$\epsilon = \frac{E_0}{S_0},$$

because most applications will involve a small ϵ. We shall have K and ϵ in mind while choosing scales such that these symbols appear naturally in the scaled equations.

Looking at the equations, we see that the K parameter will appear if $t_c \sim 1/k_+$. However, $1/k_+$ does not have the dimension $[\mathrm{T}]^{-1}$ as required, so we need to add a factor with dimension mol. A natural choice is $t_c^{-1} = k_+ S_0$ or $t_c^{-1} = k_+ E_0$. Since often $S_0 \gg E_0$, the former t_c is a short time scale and the latter is a long time scale. If the interest is in the long time scale, we set

$$t_c = \frac{1}{k_+ E_0}\,.$$

The equations then take the form

$$\begin{aligned}
\frac{d\bar{Q}}{d\bar{t}} &= \frac{S_0}{Q_c}\bar{E}\bar{S} - K E_0^{-1}\bar{Q},\\
\frac{d\bar{P}}{d\bar{t}} &= \frac{k_v}{k_+ E_0}\frac{Q_c}{P_c}\bar{Q},\\
\frac{d\bar{S}}{d\bar{t}} &= -\bar{E}\bar{S} + \frac{k_-}{k_+ E_0}\frac{Q_c}{S_0}\bar{Q},\\
\frac{d\bar{E}}{d\bar{t}} &= -\epsilon^{-1}\bar{E}\bar{S} + K\frac{Q_c}{E_0^2}\bar{Q}\,.
\end{aligned}$$

The $[ES]$ variable starts and ends at zero, and its maximum value can be roughly estimated from the equation for $[ES]'$ by setting $[ES]' = 0$, which gives

$$[ES] = \frac{[E][S]}{K} \sim \frac{E_0 S_0}{K},$$

where we have replaced $[E][S]$ by $E_0 S_0$ as an identification of magnitude. This magnitude of $[ES]$ at its maximum can be used as the characteristic size Q_c:

$$Q_c = \frac{E_0 S_0}{K}\,.$$

The equation for $\bar{P}$ simplifies if we choose $P_c = Q_c$. With these assumptions one gets

$$
\begin{aligned}
\frac{d\bar{Q}}{d\bar{t}} &= KE_0^{-1}(\bar{E}\bar{S} - \bar{Q}),\\
\frac{d\bar{P}}{d\bar{t}} &= \frac{k_v}{k_+E_0}\bar{Q},\\
\frac{d\bar{S}}{d\bar{t}} &= -\bar{E}\bar{S} + \frac{k_-}{k_+E_0}\frac{E_0}{K}\bar{Q},\\
\frac{d\bar{E}}{d\bar{t}} &= -\epsilon^{-1}\bar{E}\bar{S} + \epsilon^{-1}\bar{Q}\,.
\end{aligned}
$$

We can now identify the dimensionless numbers

$$
\alpha = \frac{K}{E_0}, \quad \beta = \frac{k_v}{k_+E_0}, \quad \gamma = \frac{k_-}{k_+E_0},
$$

where we see that $\alpha = \beta + \gamma$, so γ can be eliminated. Moreover,

$$
\alpha = \frac{k_-}{k_+E_0} + \beta,
$$

implying that $\alpha > \beta$.

We arrive at the final set of scaled differential equations:

$$
\frac{d\bar{Q}}{d\bar{t}} = \alpha(\bar{E}\bar{S} - \bar{Q}), \tag{2.57}
$$

$$
\frac{d\bar{P}}{d\bar{t}} = \beta\bar{Q}, \tag{2.58}
$$

$$
\frac{d\bar{S}}{d\bar{t}} = -\bar{E}\bar{S} + (1 - \beta\alpha^{-1})\bar{Q}, \tag{2.59}
$$

$$
\epsilon\frac{d\bar{E}}{d\bar{t}} = -\bar{E}\bar{S} + \bar{Q}\,. \tag{2.60}
$$

The initial conditions are $\bar{S} = \bar{E} = 1$ and $\bar{Q} = \bar{P} = 0$.

The five initial parameters (S_0, E_0, k_+, k_-, and k_v) are reduced to three dimensionless constants:

- α is the dimensionless Michaelis constant, reflecting the ratio of the production of P and E ($k_v + k_-$) versus the production of the complex (k_+), made dimensionless by E_0,
- ϵ is the initial fraction of enzyme relative to the substrate,
- β measures the relative importance of production of P (k_v) versus production of the complex (k_+), made dimensionless by E_0.

Observe that software developed for solving (2.47)-(2.50) cannot be reused for solving (2.57)-(2.60) since the latter system has a slightly different structure.

Conservation equations. The counterpart to the conservation equations (2.54)-(2.55) is obtained by adding (2.57) and α times (2.60), and adding (2.57), (2.58), and α times (2.59):

$$\epsilon^{-1}\alpha^{-1}\bar{Q}+\bar{E}=1, \tag{2.61}$$

$$\alpha\bar{S}+\bar{Q}+\bar{P}=\alpha\,. \tag{2.62}$$

The scaled quantities, as well as the original concentrations, must be positive variables, and $\bar{E}\in[0,1]$, $\bar{S}\in[0,1]$. Such checks along with the conserved quantities above should be performed at every time step in a simulation.

Analysis of the scaled system. In the scaled system, we may assume ϵ small, which from (2.60) gives rise to the simplification $\epsilon\bar{E}'=0$, and thereby the relation $\bar{Q}=\bar{E}\bar{S}$. The conservation equation $[ES]+[E]=E_0$ reads $Q_c\bar{Q}+E_0\bar{E}=E_0$ such that $\bar{E}=1-Q_c\bar{Q}/E_0=1-\bar{Q}S_0/K=1-\epsilon^{-1}\alpha^{-1}\bar{Q}$. The relation $\bar{Q}=\bar{E}\bar{S}$ then becomes

$$\bar{Q}=(1-\epsilon^{-1}\alpha^{-1}\bar{Q})\bar{S},$$

which can be solved for $\bar{Q}$:

$$\bar{Q}=\frac{\bar{S}}{1+\epsilon^{-1}\alpha^{-1}\bar{S}}\,.$$

The equation (2.59) for $\bar{S}$ becomes

$$\frac{d\bar{S}}{d\bar{t}}=-\beta\alpha^{-1}\bar{Q}=-\frac{\beta\bar{S}}{\alpha+\epsilon^{-1}\bar{S}}\,. \tag{2.63}$$

This is a more precise analysis than the one leading to (2.56) since we now realize that the mathematical assumption for the simplification is $\epsilon\rightarrow 0$.

Is (2.63) consistent with (2.56)? It is easy to make algebraic mistakes when deriving scaled equations, so it is always wise to carry out consistency checks. Introducing dimensions in (2.63) leads to

$$\frac{t_c}{S_0}\frac{dS}{dt}=\frac{d\bar{S}}{d\bar{t}}=-\frac{\beta\bar{S}}{\alpha+\epsilon^{-1}\bar{S}}=-\frac{k_v}{k_+E_0}\frac{S}{KE_0^{-1}+E_0^{-1}S_0\bar{S}}=-\frac{k_v}{k_+}\frac{\bar{S}}{K+S},$$

and hence with $t_c^{-1}=k_+E_0$,

$$\frac{dS}{dt}=-\frac{k_vE_0S}{K+S},$$

which is (2.56).

Figure 2.5 shows the impact of ϵ: with a moderately small value (0.1) we see that $\bar{Q}\approx 0$, which justifies the simplifications performed above. We also observe that all the unknowns vary between 0 and about 1, indicating that the scaling is successful for the chosen dimensionless numbers. The simulations made use of a time step $\Delta\bar{t} = 0.01$ with a 4th-order Runge-Kutta method, using $\alpha = 1.5$, $\beta = 1$ (relevant code is in the `simulate_biochemical_process` function in `session.py`).

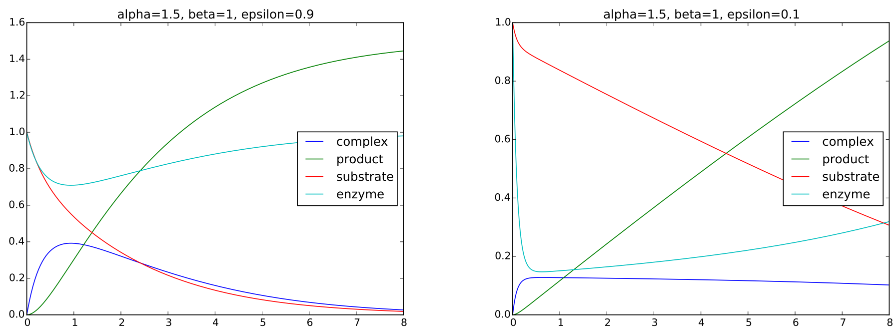


Fig. 2.5 Simulation of a biochemical process.

However, it is of interest to investigate the limit $\epsilon \to 0$. Initially, the equation for $d\bar{E}/d\bar{t}$ reads $d\bar{E}/d\bar{t} = -\epsilon^{-1}$, which implies a very fast reduction of $\bar{E}$. Using $\epsilon = 0.005$ and $\Delta\bar{t} = 10^{-3}$, simulation results show that $\bar{E}$ decays to approximately zero at $t = 0.03$ while $\bar{S} \approx 1$ and $\bar{Q} \approx \bar{P} \approx 0$. This is reasonable since with very little enzyme in comparison with the substrate ($\epsilon \to 0$) very little will happen.

2.2 Vibration problems

We shall in this section address a range of different second-order ODEs for mechanical vibrations and demonstrate how to reason about the scaling in different physical scenarios.

2.2.1 Undamped vibrations without forcing

The simplest differential equation model for mechanical vibrations reads

$$mu'' + ku = 0, \quad u(0) = I,\ u'(0) = V, \tag{2.64}$$

where unknown $u(t)$ measures the displacement of the body, This is a common model for a vibrating body with mass m attached to a linear spring with spring constant k (and force $-ku$). Figure 2.6 shows a typical mechanical sketch of such a system: some mass can move horizontally without friction and is connected to a spring that exerts a force $-ku$ on the body.

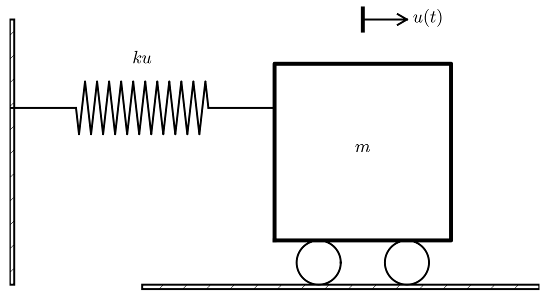


Fig. 2.6 Oscillating body attached to a spring.

The first technical steps of scaling. The problem (2.64) has one independent variable t and one dependent variable u. We introduce dimensionless versions of these variables:

$$\bar{u} = \frac{u}{u_c}, \quad \bar{t} = \frac{t}{t_c},$$

where u_c and t_c are characteristic values of u and t. Inserted in (2.64), we get

$$m\frac{u_c}{t_c^2}\frac{d^2\bar{u}}{d\bar{t}^2} + ku_c\bar{u} = 0, \quad u_c\bar{u}(0) = I, \quad \frac{u_c}{t_c}\frac{d\bar{u}}{d\bar{t}}(0) = V,$$

resulting in

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \frac{t_c^2 k}{m}\bar{u} = 0, \quad \bar{u}(0) = \frac{I}{u_c}, \ \bar{u}'(0) = \frac{Vt_c}{u_c}. \tag{2.65}$$

What is an appropriate displacement scale u_c? The initial condition $u(0) = I$ is a candidate, i.e., $u_c = I$. But how to choose the time scale? Making the coefficient in front of the $\bar{u}$ unity, such that both terms balance and are of size unity, is a candidate.

The exact solution. To better see what the proper scales of u and t are, we can look into the analytical solution of this problem. Although the exact solution of (2.64) is quite straightforward to calculate by hand, we take the opportunity to make use of SymPy to find $u(t)$. The use of SymPy can later be generalized to vibration ODEs that are harder to solve by hand.

SymPy requires all mathematical symbols to be explicitly created:

```
from sympy import *
u = symbols('u', cls=Function)
w = symbols('w', real=True, positive=True)
```

```
I, V, C1, C2 = symbols('I V C1 C2', real=True)
```

To specify the ODE to be solved, we can make a Python function returning all the terms in the ODE:

```
# Define differential equation: u'' + w**2*u = 0
def ode(u):
    return diff(u, t, t) + w**2*u

diffeq = ode(u(t))
```

The `diffeq` variable, defining the ODE, can be passed to the SymPy function `dsolve` to find the symbolic solution of the ODE:

```
s = dsolve(diffeq, u(t))
# s is an u(t) == expression (Eq obj.), s.rhs grabs the expression
u_sol = s.rhs
print u_sol
```

The solution that gets printed is `C1*sin(t*w) + C2*cos(t*w)`, indicating that there are two integration constants `C1` and `C2` to be determined by the initial conditions. The result of applying these conditions is a 2×2 linear system of algebraic equations that SymPy can solve by the `solve` function. The code goes as follows:

```
# The solution u_sol contains integration constants C1 and C2
# but these are not symbols, substitute them by symbols
u_sol = u_sol.subs('C1', C1).subs('C2', C2)

# Determine C1 and C2 from the initial conditions
ic = [u_sol.subs(t, 0) - I, u_sol.diff(t).subs(t, 0) - V]
print ic   # 2x2 algebraic system for C1 and C2
s = solve(ic, [C1, C2])
# s is now a dictionary: {C2: I, C1: V/w}
# substitute solution back in u_sol
u_sol = u_sol.subs(C1, s[C1]).subs(C2, s[C2])
print u_sol
```

The `u_sol` variable is now `I*cos(t*w) + V*sin(t*w)/w`. Since symbolic software is far from bug-free and can give wrong results, we should always check the answer. Here, we insert the solution in the ODE to see if the result is zero, and we insert the solution in the initial conditions to see that these are fulfilled:

```
# Check that the solution fulfills the ODE and init.cond.
print simplify(ode(u_sol)),
print u_sol.subs(t, 0) - I, diff(u_sol, t).subs(t, 0) - V
```

There will be many more examples on using SymPy to find exact solutions of differential equation problems.

The solution of the ODE in mathematical notation is

$$u(t) = I\cos(\omega t) + \frac{V}{\omega}\sin(\omega t), \quad \omega = \sqrt{\frac{k}{m}}\,.$$

More insight arises from rewriting such an expression in the form $A\cos(wt-\phi)$:

$$u(t)=\sqrt{I^2+\frac{V^2}{\omega^2}}\cos(wt-\phi), \quad \phi=\tan^{-1}(V/(\omega I))\,.$$

Now we see that the u corresponds to cosine oscillations with a frequency shift ϕ and amplitude $\sqrt{I^2+(V/\omega)^2}$.

The forthcoming text relies on a good understanding of concepts like period, frequency, and amplitude of oscillating signals, so readers who need to refresh these concepts are recommended to do Exercise 2.12 before continuing.

Discussion of the displacement scale. The amplitude of u is $\sqrt{I^2+V^2/\omega^2}$, and this expression is obviously a candidate for u_c. However, the simpler choice $u_c=\max(I,V/\omega)$ is also relevant and more attractive than the square root expression (but potentially a factor 1.4 wrong compared to the exact amplitude). It is not very important to have $|u|\leq 1$, the point is to avoid $|u|$ very small or large.

Discussion of the time scale. What is an appropriate time scale? Looking at (2.65) and arguing that $\bar{u}''$ and $\bar{u}$ both should be around unity in size, the coefficient t_c^2k/m must equal unity, implying that $t_c=\sqrt{m/k}$. Also from the analytical solution we see that the solution goes like the sine or cosine of ωt, so $1/\omega=\sqrt{m/k}$ can be a characteristic time scale. Likewise, one period of the oscillations, $P=2\pi/\omega$, can be the characteristic time, leading to $t_c=2\pi/\omega$.

The dimensionless solution. With $u_c=I$ and $t_c=\sqrt{m/k}$ we get the scaled model

$$\frac{d^2\bar{u}}{d\bar{t}^2}+\bar{u}=0, \quad \bar{u}(0)=1,\ \bar{u}'(0)=\alpha, \tag{2.66}$$

where α is a dimensionless parameter:

$$\alpha=\frac{V}{I}\sqrt{\frac{m}{k}}\,.$$

Note that in case $V=0$, we have “scaled away” all physical parameters. The universal solution without physical parameters is then $\bar{u}(\bar{t})=\cos\bar{t}$.

The unscaled solution is recovered as

$$u(t)=I\bar{u}(\sqrt{k/m}\bar{t})\,. \tag{2.67}$$

This expressions shows that the scaling is simply a matter of *stretching or shrinking the axes*.

Alternative displacement scale. Using $u_c=V/\omega$, the equation is not changed, but the initial conditions become

$$\bar{u}(0) = \frac{I}{u_c} = \frac{I\omega}{V} = \frac{I}{V}\sqrt{\frac{k}{m}} = \alpha^{-1}, \quad \bar{u}'(0) = 1\,.$$

With $u_c = V/\omega$ and one period as time scale, $t_c = 2\pi\sqrt{m/k}$, we get the alternative model

$$\frac{d^2\bar{u}}{d\bar{t}^2} + 4\pi^2\bar{u} = 0, \quad \bar{u}(0) = \alpha^{-1},\ \bar{u}'(0) = 2\pi\,. \tag{2.68}$$

The unscaled solution is in this case recovered by

$$u(t) = V\sqrt{\frac{m}{k}}\bar{u}(2\pi\sqrt{k/m}\bar{t})\,. \tag{2.69}$$

About frequency and dimensions. The solution goes like $\cos\omega t$, where $\omega = \sqrt{m/k}$ must have dimension 1/s. Actually, ω has dimension *radians per second*: rad/s. A radian is dimensionless since it is arc (length) divided by radius (length), but still regarded as a unit. The period P of vibrations is a more intuitive quantity than the frequency ω. The relation between P and ω is $P = 2\pi/\omega$. The number of oscillation cycles per period, f, is a more intuitive measurement of frequency and also known as *frequency*. Therefore, to be precise, ω should be named *angular frequency*. The relation between f and T is $f = 1/T$, so $f = 2\pi\omega$ and measured in Hz (1/s), which is the unit for counts per unit time.

2.2.2 Undamped vibrations with constant forcing

For vertical vibrations in the gravity field, the model (2.64) must also take the gravity force $-mg$ into account:

$$mu'' + ku = -mg\,.$$

How does the new term $-mg$ influence the scaling? We observe that if there is no movement of the body, $u'' = 0$, and the spring elongation matches the gravity force: $ku = -mg$, leading to a steady displacement $u = -mg/k$. We can then have oscillations around this equilibrium point. A natural scaling for u is therefore

$$\bar{u} = \frac{u - (-mg/k)}{u_c} = \frac{uk + mg}{ku_c}\,.$$

The scaled differential equation with the same time scale as before reads

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \bar{u} - \frac{t_c^2}{u_c}g = -\frac{t_c^2}{u_c}g,$$

leading to

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \bar{u} = 0\,.$$

The initial conditions $u(0) = I$ and $u'(0) = V$ become, with $u_c = I$,

$$\bar{u}(0) = 1 + \frac{mg}{kI}, \quad \frac{d\bar{u}}{d\bar{t}}(0) = \sqrt{\frac{m}{k}}\frac{V}{I}\,.$$

We see that the oscillations around the equilibrium point in the gravity field are identical to the horizontal oscillations without gravity, except for an offset $mg/(kI)$ in the displacement.

2.2.3 Undamped vibrations with time-dependent forcing

Now we add a transient forcing term $F(t)$ to the model (2.64):

$$mu'' + ku = F(t), \quad u(0) = I,\ u'(0) = V\,. \tag{2.70}$$

Take the forcing to be oscillating:

$$F(t) = A\cos(\psi t)\,.$$

The technical steps of the scaling are still the same, with the intermediate result

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \frac{t_c^2 k}{m}\bar{u} = \frac{t_c^2}{mu_c}A\cos(\psi t_c\bar{t}), \quad \bar{u}(0) = \frac{I}{u_c},\ \bar{u}'(0) = \frac{Vt_c}{u_c}\,. \tag{2.71}$$

What are typical displacement and time scales? This is not so obvious without knowing the details of the solution, because there are three parameters (I, V, and A) that influence the magnitude of u. Moreover, there are two time scales, one for the free vibrations of the systems and one for the forced vibrations $F(t)$.

Investigating scales via analytical solutions. As we have seen already several times, having access to an exact solution is very fortunate as it allows us to directly examine the scales. Also in the present problem it is possible to derive an exact solution. We continue the SymPy session from the previous section and perform much of the same steps. Note that we use `w` for $\omega = \sqrt{k/m}$ in the computer code (to obtain a more direct visual counterpart to ω). SymPy may get confused when coefficients in differential equations contain several symbols. We therefore rewrite the equation with at most one symbol in each coefficient (i.e., symbolic software is in general more successful when applied to scaled differential equations than the unscaled counterparts, but right now our task is to solve the unscaled version). The amplitude A/m in the forcing term is of this reason replaced by the symbol `A1`.

```
A, A1, m, psi = symbols('A A1 m psi', positive=True, real=True)
def ode(u):
    return diff(u, t, t) + w**2*u - A1*cos(psi*t)

diffeq = ode(u(t))
u_sol = dsolve(diffeq, u(t))
u_sol = u_sol.rhs

# Determine the constants C1 and C2 in u_sol
# (first substitute our own declared C1 and C2 symbols,
# then use the initial conditions)
u_sol = u_sol.subs('C1', C1).subs('C2', C2)
eqs = [u_sol.subs(t, 0) - I, u_sol.diff(t).subs(t, 0) - V]
s = solve(eqs, [C1, C2])
u_sol = u_sol.subs(C1, s[C1]).subs(C2, s[C2])

# Check that the solution fulfills the equation and init.cond.
print simplify(ode(u_sol))
print simplify(u_sol.subs(t, 0) - I)
print simplify(diff(u_sol, t).subs(t, 0) - V)

u_sol = simplify(expand(u_sol.subs(A1, A/m)))
print u_sol
```

The output from the last line is

```
A/m*cos(psi*t)/(-psi**2 + w**2) + V*sin(t*w)/w +
(A/m + I*psi**2 - I*w**2)*cos(t*w)/(psi**2 - w**2)
```

With a bit of rewrite this expression becomes

$$u(t) = \frac{A/m}{\omega^2 - \psi^2}\cos(\psi t) + \frac{V}{\omega}\sin(\omega t) + \left(\frac{A/m}{\psi^2 - \omega^2} + I\right)\cos(\omega t)\,.$$

Obviously, this expression is only meaningful for $\psi \neq \omega$. The case $\psi = \omega$ gives an infinite amplitude in this model, a phenomenon known as resonance. The amplitude becomes finite when damping is included, see Section 2.2.4.

When the system starts from rest, $I = V = 0$, and the forcing is the only driving mechanism, we can simplify:

$$\begin{aligned} u(t) &= \frac{A}{m(\omega^2 - \psi^2)}\cos(\psi t) + \frac{A}{m(\psi^2 - \omega^2)}\cos(\omega t) \\ &= \frac{A}{m(\omega^2 - \psi^2)}(\cos(\psi t) - \cos(\omega t))\,. \end{aligned}$$

To gain more insight, $\cos(\psi t) - \cos(\omega t)$ can be rewritten in terms of the mean frequency $(\psi + \omega)/2$ and the difference in frequency $(\psi - \omega)/2$:

$$u(t) = \frac{A}{m(\omega^2 - \psi^2)} 2\sin\left(\frac{\psi - \omega}{2}t\right)\sin\left(\frac{\psi + \omega}{2}t\right), \tag{2.72}$$

showing that there is a signal with frequency $(\psi+\omega)/2$ whose amplitude has a (much) slower frequency $(\psi-\omega)/2$. Figure 2.7 shows an example on such a signal.

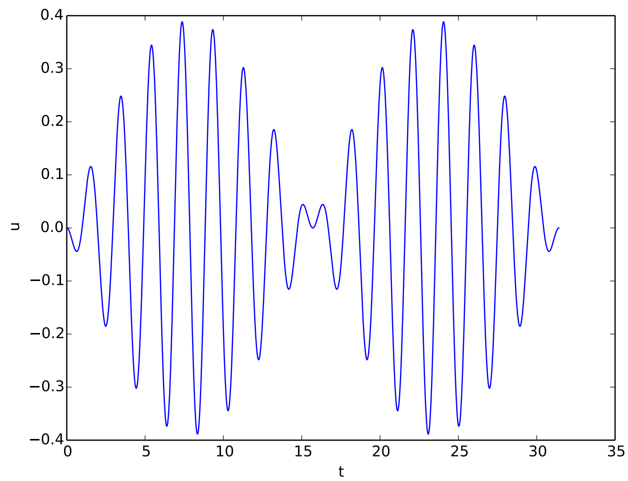


Fig. 2.7 Signal with frequency 3.1 and envelope frequency 0.2.

The displacement and time scales. A characteristic displacement can in the latter special case be taken as $u_c = A/(m(\omega^2 - \psi^2))$. This is also a relevant choice in the more general case $I \neq 0, V \neq 0$, unless I or V is so large that it dominates over the amplitude caused by the forcing. With $u_c = A/(m(\omega^2 - \psi^2))$ we also have three special cases: $\omega \ll \psi$, $\omega \gg \psi$, and $\psi \sim \omega$. In the latter case we need $u_c = A/(m(\omega^2 - \psi^2))$ if we want $|u| \leq 1$. When ω and ψ are significantly different, we may choose one of them and neglect the smaller. Choosing ω means $u_c = A/k$, which is the relevant scale if $\omega \gg \psi$. In the opposite case, $\omega \ll \psi$, $u_c = A/(m\psi^2)$.

The time scale is dominated by the fastest oscillations, which are of frequency ψ or ω when these are close and the largest of them when they are distant. In any case, we set $t_c = 1/\max(\psi,\omega)$.

Finding the displacement scale from the differential equation. Going back to (2.71), we may demand that all the three terms in the differential equation are of size unity. This leads to $t_c = \sqrt{m/k}$ and $u_c = At_c^2/m = A/k$. The formula for u_c is a kind of measure of the ratio of the forcing and the spring force (the dimensionless number $A/(ku_c)$ would be this ratio).

Looking at (2.72), we see that if $\psi \ll \omega$, the amplitude can be approximated by $A/(m\omega^2) = A/k$, showing that the scale $u_c = A/k$ is relevant for an excitation frequency ψ that is small compared to the free vibration frequency ω.

Scaling with free vibrations as time scale. The next step is to work out the dimensionless ODE for the chosen scales. We first select the time scale based on the free oscillations with frequency ω, i.e., $t_c = 1/\omega$. Inserting the expression in (2.71) results in

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \bar{u} = \gamma\cos(\delta\bar{t}), \quad \bar{u}(0) = \alpha,\ \bar{u}'(0) = \beta\,. \tag{2.73}$$

Here we have four dimensionless variables

$$\alpha = \frac{I}{u_c}, \tag{2.74}$$

$$\beta = \frac{Vt_c}{u_c} = \frac{V}{\omega u_c}, \tag{2.75}$$

$$\gamma = \frac{t_c^2 A}{m u_c} = \frac{A}{k u_c}, \tag{2.76}$$

$$\delta = \frac{t_c}{\psi^{-1}} = \frac{\psi}{\omega}\,. \tag{2.77}$$

We remark that the choice of u_c has so far not been made. Several different cases will be considered below, and we will see that certain choices reduce the number of independent dimensionless variables to three.

The four dimensionless variables above have interpretations as ratios of physical effects:

- α: ratio of the initial displacement and the characteristic response u_c,
- β: ratio of the initial velocity and the typical velocity measure u_c/t_c,
- γ: ratio of the forcing A and the mass times acceleration mu_c/t_c^2 *or* the ratio of the forcing and the spring force ku_c
- δ: ratio of the frequencies or the time scales of the forcing and the free vibrations.

Software. Any solver for (2.71) can be used for (2.73). More details are provided at the end of Section 2.2.4.

Choice of u_c close to resonance. Now we shall discuss various choices of u_c. Close to resonance, when $\psi \sim \omega$, we may set $u_c = A/(m(\omega^2 - \psi^2))$. The dimensionless numbers become in this case

$$
\begin{aligned}
\alpha &= \frac{I}{u_c} = \frac{I}{A/k}(1-\delta^2),\\
\beta &= \frac{V}{\omega u_c} = \frac{V\sqrt{km}}{A}(1-\delta^2),\\
\gamma &= \frac{A}{ku_c} = 1-\delta^2,\\
\delta &= \frac{\psi}{\omega}\,.
\end{aligned}
$$

With $\psi = 0.99\omega$, $\delta = 0.99$, $V = 0$, $\alpha = \gamma = 1-\delta^2 = 0.02$, we have the problem

$$
\frac{d^2\bar{u}}{d\bar{t}^2} + \bar{u} = 0.02\cos(0.99\bar{t}), \quad \bar{u}(0) = 0.02,\ \bar{u}'(0) = 0\,.
$$

This is a problem with a very small initial condition and a very small forcing, but the state close to resonance brings the amplitude up to about unity, see the result of numerical simulations with $\delta = 0.99$ in Figure 2.8. Neglecting α, the solution is given by (2.72), which here means $A = 1-\delta^2$, $m = 1$, $\omega = 1$, $\psi = \delta$:

$$
\bar{u}(\bar{t}) = 2\sin(-0.005\bar{t})\sin(0.995\bar{t})\,.
$$

Note that this is a problem which demands very high accuracy in the numerical calculations. Using 20 time steps per period gives a significant angular frequency error and an amplitude of about 1.4. We used 160 steps per period for the results in Figure 2.8.

Unit size of all terms in the ODE. Using the displacement scale $u_c = A/k$ leads to (2.73) with

$$
\begin{aligned}
\alpha &= \frac{I}{u_c} = \frac{I}{A/k},\\
\beta &= \frac{V}{\omega u_c} = \frac{Vk}{A\omega},\\
\gamma &= \frac{A}{ku_c} = 1,\\
\delta &= \frac{\psi}{\omega}\,.
\end{aligned}
$$

Simulating a case with $\delta = 0.5$, $\alpha = 1$, and $\beta = 0$ gives the oscillations in Figure 2.9, which is a case away from resonance, and the amplitude is about unity. However, choosing $\delta = 0.99$ (close to resonance) results in a figure similar to Figure 2.8, except that the amplitude is about 10^2 because of the moderate size of u_c. The present scaling is therefore most suitable away from resonance, and when the terms containing $\cos\omega t$ and $\sin\omega t$ are important (e.g., $\omega \gg \psi$).

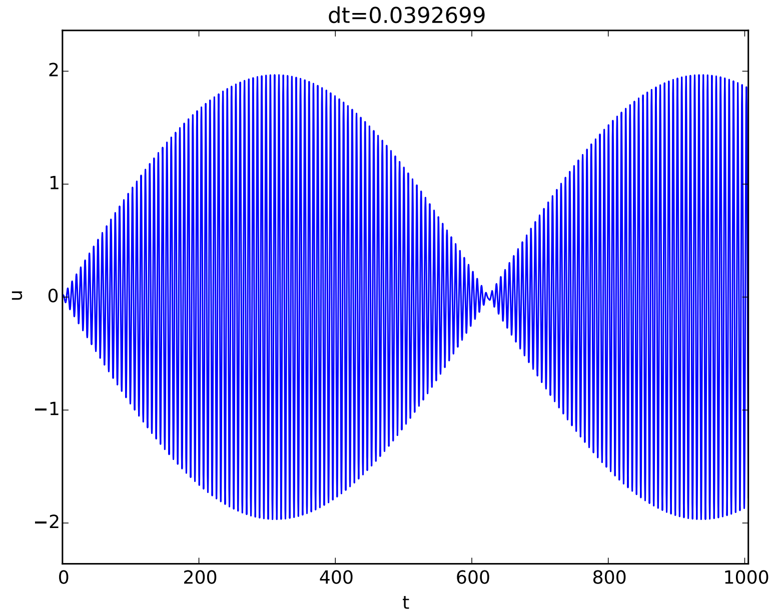


Fig. 2.8 Forced undamped vibrations close to resonance.

Choice of u_c when $\psi \gg \omega$. Finally, we may look at the case where $\psi \gg \omega$ such that $u_c = A/(m\psi^2)$ is a relevant scale (i.e., omitting ω^2 compared to ψ^2 in the denominator), but in this case we should use $t_c = 1/\psi$ since the force varies much faster than the free vibrations of the system. This choice of t_c changes the scaled ODE to

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \delta^{-2}\bar{u} = \gamma\cos(\bar{t}), \quad \bar{u}(0) = \alpha,\ \bar{u}'(0) = \beta, \tag{2.78}$$

where

$$\begin{aligned}
\alpha &= \frac{I}{u_c} = \frac{I}{A/k}\delta^2,\\
\beta &= \frac{Vt_c}{u_c} = \frac{V\sqrt{km}}{A}\delta,\\
\gamma &= \frac{t_c^2 A}{mu_c} = 1,\\
\delta &= \frac{t_c}{\psi^{-1}} = \frac{\psi}{\omega}\,.
\end{aligned}$$

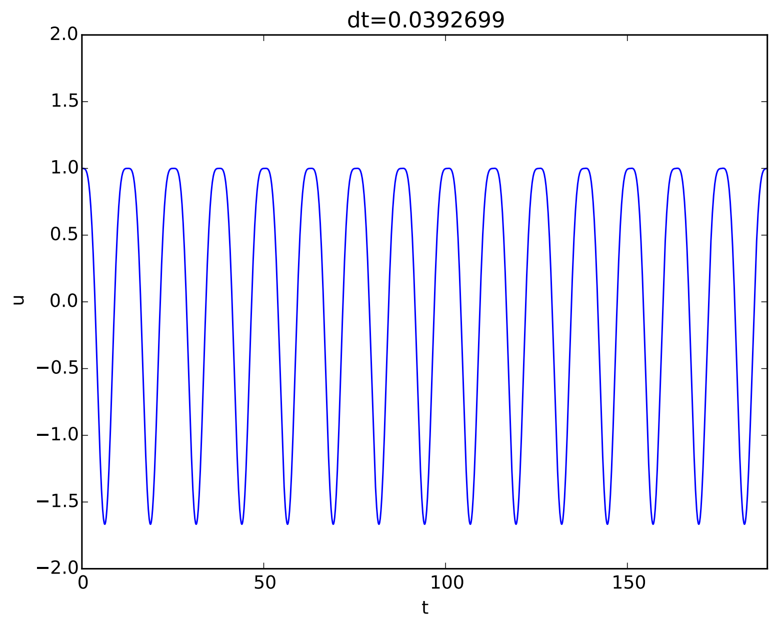


Fig. 2.9 Forced undamped vibrations away from resonance.

In the regime $\psi \gg \omega$, $\delta \gg 1$, thus making α and β large. However, if α and/or β is large, the initial condition dominates over the forcing, and will also dominate the amplitude of u, thereby making the scaling of u inappropriate. In case $I = V = 0$ so that $\alpha = \beta = 0$, (2.72) predicts ($A = m = 1$, $\omega = \delta^{-1}$, $\psi = 1$)

$$\bar{u}(\bar{t}) = (\delta^{-2} - 1)^{-1} 2 \sin\left(\frac{1}{2}(1 - \delta^{-1})\bar{t}\right) \sin\left(\frac{1}{2}(1 + \delta^{-1})\bar{t}\right),$$

which has an amplitude about 2 for $\delta \gg 1$. Figure 2.10 shows a case.

With $\alpha = 0.05\delta^2 = 5$, we get a significant contribution from the free vibrations (the homogeneous solution of the ODE) as shown in Figure 2.11. For larger α values, one must base u_c on I instead. (The graphs in Figure 2.10 and 2.11 were produced by numerical simulations with 160 time steps per period of the forcing.)

Displacement scale based on I. Choosing $u_c = I$ gives

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \bar{u} = \gamma \cos(\delta \bar{t}), \quad \bar{u}(0) = 1,\ \bar{u}'(0) = \beta, \tag{2.79}$$

with

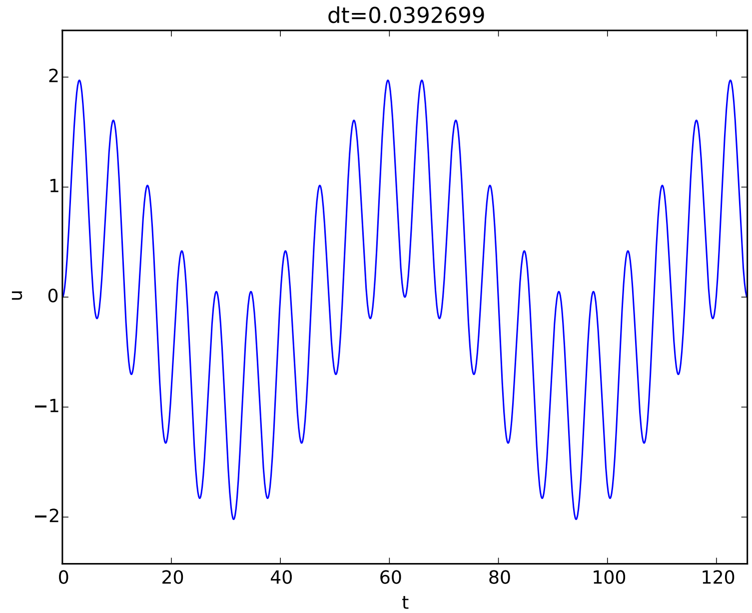


Fig. 2.10 Forced undamped vibrations with rapid forcing.

$$\beta = \frac{Vt_c}{u_c} = \frac{V}{I}\sqrt{\frac{m}{k}}, \tag{2.80}$$

$$\gamma = \frac{tc^2 A}{mu_c} = \frac{A}{ku_c} = \frac{A}{kI}\,. \tag{2.81}$$

This scaling is not relevant close to resonance since then $u_c \gg I$.

2.2.4 Damped vibrations with forcing

We now introduce a linear damping force $bu'(t)$ in the equation of motion:

$$mu'' + bu' + ku = A\cos(\psi t), \quad u(0) = I,\ u'(0) = V\,. \tag{2.82}$$

Figure 2.12 shows a typical one-degree-of-freedom mechanical system with a linear dashpot, representing the damper (bu'), a linear spring (ku), and an external force (F).

The standard scaling procedure results in

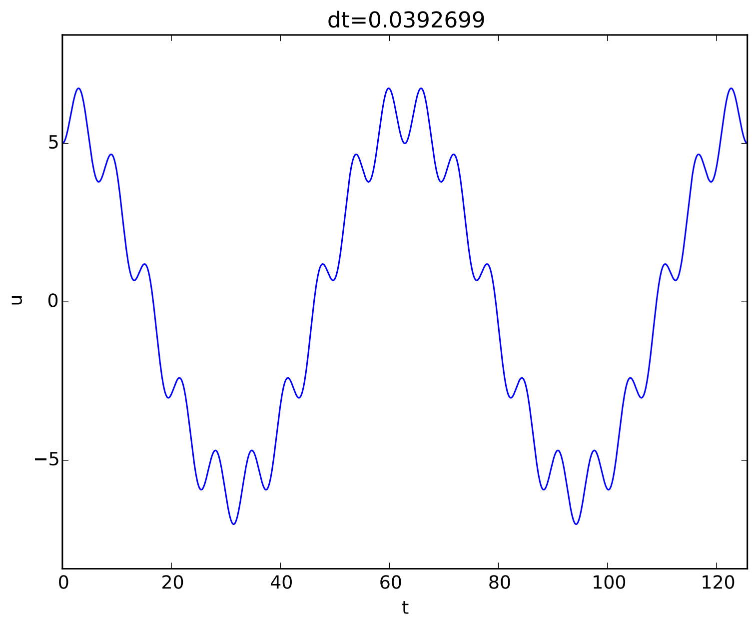


Fig. 2.11 Forced undamped vibrations with rapid forcing and initial displacement of 5.

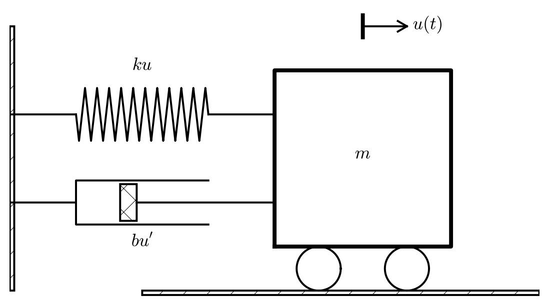


Fig. 2.12 Oscillating body with external force, attached to a spring and damper.

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \frac{t_c b}{m}\frac{d\bar{u}}{d\bar{t}} + \frac{t_c^2 k}{m}\bar{u} = \frac{t_c^2}{m u_c}A\cos(\psi t_c \bar{t}), \quad \bar{u}(0) = \frac{I}{u_c},\ \bar{u}'(0) = \frac{V t_c}{u_c}. \quad (2.83)$$

The exact solution. As always, it is a great advantage to look into exact solutions for controlling our choice of scales. Using SymPy to solve (2.82) is, in principle, very straightforward:

```
>>> diffeq = diff(u(t), t, t) + b/m*diff(u(t), t) + w**2*u(t)
>>> s = dsolve(diffeq, u(t))
>>> s.rhs
```

```
C1*exp(t*(-b - sqrt(b - 2*m*w)*sqrt(b + 2*m*w))/(2*m)) +
C2*exp(t*(-b + sqrt(b - 2*m*w)*sqrt(b + 2*m*w))/(2*m))
```

This is indeed the correct solution, but it is on a complex exponential function form, valid for all b, m, and ω. We are interested in the case with *small damping*, $b < 2m\omega$, where the solution is an exponentially damped sinusoidal function. Rewriting the expression in the right form is tricky with SymPy commands. Instead, we demonstrate a common technique when doing symbolic computing: general procedures like `dsolve` are replaced by manual steps. That is, we solve the ODE "by hand", but use SymPy to assist the calculations.

The solution is composed of a homogeneous solution u_h of $mu'' + bu' + ku = 0$ and one particular solution u_p of the nonhomogeneous equation $mu'' + bu' + ku = A\cos(\psi t)$. The homogeneous solution with damped oscillations (requiring $b < 2\sqrt{mk}$) can be found by the following code. We have divided the differential equation by m and introduced $B = \frac{1}{2}b/m$ and let `A1` represent A/m to simplify expressions and help SymPy with less symbols in the equation. Without these simplifications, SymPy stalls in the computations due to too many symbols in the equation. The problem is actually a solid argument for scaling differential equations before asking SymPy to solve them since scaling effectively reduces the number of parameters in the equations!

The following SymPy steps derives the solution of the homogeneous ODE:

```
u = symbols('u', cls=Function)
t, w, B, A, A1, m, psi = symbols('t w B A A1 m psi',
                                 positive=True, real=True)

def ode(u, homogeneous=True):
    h = diff(u, t, t) + 2*B*diff(u, t) + w**2*u
    f = A1*cos(psi*t)
    return h if homogeneous else h - f

# Find coefficients in polynomial (in r) for exp(r*t) ansatz
r = symbols('r')
ansatz = exp(r*t)
poly = simplify(ode(ansatz)/ansatz)

# Convert to polynomial to extract coefficients
poly = Poly(poly, r)
# Extract coefficients in poly: a_*t**2 + b_*t + c_
a_, b_, c_ = poly.coeffs()
# Assume b_**2 - 4*a_*c_ < 0
d = -b_/(2*a_)
if a_ == 1:
    omega = sqrt(c_ - (b_/2)**2)  # nicer formula
else:
    omega = sqrt(4*a_*c_ - b_**2)/(2*a_)

# The homogeneous solution is a linear combination of a
# cos term (u1) and a sin term (u2)
```

```
u1 = exp(d*t)*cos(omega*t)
u2 = exp(d*t)*sin(omega*t)
C1, C2, V, I = symbols('C1 C2 V I', real=True)
u_h = simplify(C1*u1 + C2*u2)
print 'u_h:', u_h
```

The print out shows

$$u_h = e^{-Bt}\left(C_1\cos(\sqrt{\omega^2 - B^2}t) + C_2\sin(\sqrt{\omega^2 - B^2}t)\right),$$

where C_1 and C_2 must be determined by the initial conditions later. It is wise to check that u_h is indeed a solution of the homogeneous differential equation:

```
assert simplify(ode(u_h)) == 0
```

We have previously just printed the residuals of the ODE and initial conditions after inserting the solution, but it is better in a code to let the programming language test that the residuals are symbolically zero. This is achieved using the `assert` statement in Python. The argument is a boolean expression, and if the expression evaluates to `False`, an `AssertionError` is raised and the program aborts (otherwise `assert` runs silently for a `True` boolean expression). Hereafter, we will use `assert` for consistency checks in computer code.

The ansatz for the particular solution u_p is

$$u_p = C_3\cos(\psi t) + C_4\sin(\psi t),$$

which inserted in the ODE gives two equations for C_3 and C_4. The relevant SymPy statements are

```
# Particular solution
C3, C4 = symbols('C3 C4')
u_p = C3*cos(psi*t) + C4*sin(psi*t)
eqs = simplify(ode(u_p, homogeneous=False))

# Collect cos(omega*t) terms
print 'eqs:', eqs
eq_cos = simplify(eqs.subs(sin(psi*t), 0).subs(cos(psi*t), 1))
eq_sin = simplify(eqs.subs(cos(psi*t), 0).subs(sin(psi*t), 1))
s = solve([eq_cos, eq_sin], [C3, C4])
u_p = simplify(u_p.subs(C3, s[C3]).subs(C4, s[C4]))

# Check that the solution is correct
assert simplify(ode(u_p, homogeneous=False)) == 0
```

Using the initial conditions for the complete solution $u = u_h + u_p$ determines C_1 and C_2:

```
u_sol = u_h + u_p  # total solution
# Initial conditions
eqs = [u_sol.subs(t, 0) - I, u_sol.diff(t).subs(t, 0) - V]
```

```
# Determine C1 and C2 from the initial conditions
s = solve(eqs, [C1, C2])
u_sol = u_sol.subs(C1, s[C1]).subs(C2, s[C2])
```

Finally, we should check that `u_sol` is indeed the correct solution:

```
checks = dict(
    ODE=simplify(expand(ode(u_sol, homogeneous=False))),
    IC1=simplify(u_sol.subs(t, 0) - I),
    IC2=simplify(diff(u_sol, t).subs(t, 0) - V))
for check in checks:
    msg = '%s residual: %s' % (check, checks[check])
    assert checks[check] == sympify(0), msg
```

Finally, we may take `u_sol = u_sol.subs(A, A/m)` to get the right expression for the solution. Using `latex(u_sol)` results in a huge expression, which should be manually ordered to something like the following:

$$
\begin{aligned}
u = &\frac{Am^{-1}}{4B^2\psi^2+\Omega^2}\left(2B\psi\sin(\psi t)-\Omega\cos(\psi t)\right)+\\
&e^{-Bt}\left(C_1\cos\left(t\sqrt{\omega^2-B^2}\right)+C_2\sin\left(t\sqrt{\omega^2-B^2}\right)\right)\\
C_1 &= \frac{Am^{-1}\Omega+4IB^2\psi^2+I\Omega^2}{4B^2\psi^2+\Omega^2}\\
C_2 &= \frac{-Am^{-1}B\Omega+4IB^3\psi^2+IB\Omega^2+4VB^2\psi^2+V\Omega^2}{\sqrt{\omega^2-B^2}\left(4B^2\psi^2+\Omega^2\right)},\\
\Omega &= \psi^2-\omega^2\,.
\end{aligned}
$$

The most important feature of this solution is that there are two time scales with frequencies ψ and $\sqrt{\omega^2-B^2}$, respectively, but the latter appears in terms that decay as e^{-Bt} in time. The attention is usually on longer periods of time, so in that case the solution simplifies to

$$
\begin{aligned}
u &= \frac{Am^{-1}}{4B^2\psi^2+\Omega^2}\left(2B\psi\sin(\psi t)-\Omega\cos(\psi t)\right)\\
&= \frac{A}{m}\frac{1}{\sqrt{4B^2\psi^2+\Omega^2}}\cos(\psi t+\phi)\frac{(\psi\omega)^{-1}}{(\psi\omega)^{-1}}\\
&= \frac{A}{k}Q\delta^{-1}\left(1+Q^2(\delta-\delta^{-1})\right)^{-\frac{1}{2}}\cos(\psi t+\phi),
\end{aligned}
\tag{2.84}
$$

where we have introduced the dimensionless numbers

$$
Q=\frac{\omega}{2B},\quad \delta=\frac{\psi}{\omega},
$$

and

$$\phi = \tan^{-1}\left(-\frac{2B}{\omega^2-\psi^2}\right) = \tan^{-1}\left(\frac{Q^{-1}}{\delta^2-1}\right).$$

Q is commonly called *quality factor* and ϕ is the *phase shift*. Dividing (2.84) by A/k, which is a common scale for u, gives the dimensionless relation

$$\frac{u}{A/k} = \frac{Q}{\delta}R(Q,\delta)^{\frac{1}{2}}\cos(\psi t+\phi), \quad R(Q,\delta) = \left(1+Q^2(\delta-\delta^{-1})\right)^{-1}. \tag{2.85}$$

Choosing scales. Much of the discussion about scales in the previous sections are relevant also when damping is included. Although the oscillations with frequency $\sqrt{\omega^2-B^2}$ die out for $t \gg B^{-1}$, we start with using this frequency for the time scale. A highly relevant assumption for engineering applications of (2.82) is that the damping is small. Therefore, $\sqrt{\omega^2-B^2}$ is close to ω and we simply apply $t_c = 1/\omega$ as before (if not the interest in large t for which the oscillations with frequency ω has died out).

The coefficient in front of the $\bar{u}'$ term then becomes

$$\frac{b}{m\omega} = \frac{2B}{\omega} = Q^{-1}.$$

The rest of the ODE is given in the previous section, and the particular formulas depend on the choices of t_c and u_c.

Choice of u_c at resonance. The relevant scale for u_c at or nearby resonance ($\psi=\omega$) becomes different from the previous section, since with damping, the maximum amplitude is a finite value. For $t \gg B^{-1}$, when the $\sin\psi t$ term is dominating, we have for $\psi=\omega$:

$$u = \frac{Am^{-1}2B\psi}{4B^2\psi^2}\sin(\psi t) = \frac{A}{2Bm\psi}\sin(\psi t) = \frac{A}{b\psi}\sin(\psi t).$$

This motivates the choice

$$u_c = \frac{A}{b\psi} = \frac{A}{b\omega}.$$

(It is wise during computations like this to stop and check the dimensions: A must be [MLT^{-2}] from the original equation ($F(t)$ must have the same dimension as mu''), bu' must also have dimension [MLT^{-2}], implying that b has dimension [MT^{-1}]. A/b then has dimension LT^{-1}, and $A/(b\psi)$ gets dimension $[L]$, which matches what we want for u_c.)

The differential equation on dimensionless form becomes

$$\frac{d^2\bar{u}}{d\bar{t}^2} + Q^{-1}\frac{d\bar{u}}{d\bar{t}} + \bar{u} = \gamma\cos(\delta\bar{t}), \quad \bar{u}(0)=\alpha,\ \bar{u}'(0)=\beta, \tag{2.86}$$

with

$$\alpha = \frac{I}{u_c} = \frac{Ib}{A}\sqrt{\frac{k}{m}}, \tag{2.87}$$

$$\beta = \frac{Vt_c}{u_c} = \frac{Vb}{A}, \tag{2.88}$$

$$\gamma = \frac{t_c^2 A}{mu_c} = \frac{b\omega}{k}, \tag{2.89}$$

$$\delta = \frac{t_c}{\psi^{-1}} = \frac{\psi}{\omega} = 1\,. \tag{2.90}$$

Choice of u_c when $\omega \gg \psi$. In the limit $\omega \gg \psi$ and $t \gg B^{-1}$,

$$u \approx \frac{A}{m\omega^2}\cos\psi t = \frac{A}{k}\cos\psi t,$$

showing that $u_c = A/k$ is an appropriate displacement scale. (Alternatively, we get this scale also from demanding $\gamma = 1$ in the ODE.) The dimensionless numbers α, β, and δ are as for the forced vibrations without damping.

Choice of u_c when $\omega \ll \psi$. In the limit $\omega \ll \psi$, we should base t_c on the rapid variations in the excitation: $t_c = 1/\psi$.

Software. It is easy to reuse a solver for a general vibration problem also in the dimensionless case. In particular, we may use the `solver` function in the file `vib.py`:

```
def solver(I, V, m, b, s, F, dt, T, damping='linear'):
```

for solving the ODE problem

$$mu'' + f(u') + s(u) = F(t), \quad u(0) = I,\ u'(0) = V,\ t \in (0,T],$$

with time steps `dt`. With `damping='linear'`, we have $f(u') = bu'$, while the other value is `'quadratic'`, meaning $f(u') = b|u'|u'$. Given the dimensionless numbers α, β, γ, δ, and Q, an appropriate call for solving (2.73) is

```
u, t = solver(I=alpha, V=beta, m=1, b=1.0/Q,
              s=lambda u: u, F=lambda t: gamma*cos(delta*t),
        dt=2*pi/n, T=2*pi*P)
```

where `n` is the number of intervals per period and `P` is the number of periods to be simulated. We way wrap this call in a `solver_scaled` function and wrap it furthermore with `joblib` to avoid repeated calls, as we explained in Section 2.1.4:

```
from vib import solver as solver_unscaled

def solver_scaled(alpha, beta, gamma, delta, Q, T, dt):
    """
    Solve u'' + (1/Q)*u' + u = gamma*cos(delta*t),
    u(0)=alpha, u'(1)=beta, for (0,T] with step dt.
    """
```

```
    print 'Computing the numerical solution'
    from math import cos
    return solver_unscaled(I=alpha, V=beta, m=1, b=1./Q,
                           s=lambda u: u,
                           F=lambda t: gamma*cos(delta*t),
                           dt=dt, T=T, damping='linear')

import joblib
disk_memory = joblib.Memory(cachedir='temp')
solver_scaled = disk_memory.cache(solver_scaled)
```

This code is found in `vib_scaled.py` and features an application for running the scaled problem with options on the command-line for α, β, γ, δ, Q, number of time steps per period, and number of periods (see the `main` function). It is an ideal application for exploring scaled vibration models.

2.2.5 Oscillating electric circuits

The differential equation for an oscillating electric circuit is very similar to the equation for forced, damped, mechanical vibrations, and their dimensionless form is identical. This fact will now be demonstrated.

The current $I(t)$ in a circuit having an inductor with inductance L, a capacitor with capacitance C, and overall resistance R, obeys the equation

$$\ddot{I} + \frac{R}{L}\dot{I} + \frac{1}{LC}I = V(t), \tag{2.91}$$

where $V(t)$ is the voltage source powering the circuit. We introduce

$$\bar{I} = \frac{I}{I_c}, \quad \bar{t} = \frac{t}{t_c},$$

and get

$$\frac{d^2\bar{I}}{d\bar{t}^2} + \frac{t_cR}{L}\frac{d\bar{I}}{d\bar{t}} + \frac{t_c^2}{LC}\bar{I} = \frac{t_c^2V_c}{I_c}\bar{V}(t)\,.$$

Here, we have scaled $V(t)$ according to

$$\bar{V}(\bar{t}) = \frac{V(t_c\bar{t})}{\max_t V(t)}\,.$$

The time scale t_c is chosen to make $\ddot{I}$ and $I/(LC)$ balance, $t_c = \sqrt{LC}$. Choosing I_c to make the coefficient in the source term of unit size, means $I_c = LCV_c$. With

$$Q^{-1} = R\sqrt{\frac{C}{L}},$$

we get the scaled equation

$$\frac{d^2\bar{I}}{d\bar{t}^2} + Q^{-1}\frac{d\bar{I}}{d\bar{t}} + \bar{I} = \bar{V}(t), \tag{2.92}$$

which is basically the same as we derived for mechanical vibrations. (Two additional dimensionless variables will arise from the initial conditions for I, just as in the mechanics cases.)

2.3 Exercises

Exercise 2.1: Perform unit conversion

Density (mass per volume: $[\mathrm{ML}^{-3}]$) of water is given as 1.05 ounce per fluid ounce. Use the `PhysicalQuantity` object to convert to $\mathrm{kg\,m}^{-3}$.

Solution. Use `pydoc PhysicalQuantities` to find that `floz` is the name of the volume "fluid ounce" and `oz` is the name of the mass "ounce". Here is an interactive session for the conversion:

```
>>> from PhysicalQuantities import PhysicalQuantity as PQ
>>> d = PQ('1.05 oz/floz')
>>> d.convertToUnit('kg/m**3')
>>> print d
1006.54198946 kg/m**3
```

Filename: `density_conversion`.

Problem 2.2: Scale a simple formula

The height y of a body thrown up in the air is given by

$$y = v_0 t - \frac{1}{2}gt^2,$$

where t is time, v_0 is the initial velocity of the body at $t=0$, and g is the acceleration of gravity. Scale this formula. Use two choices of the characteristic time: the time it takes to reach the maximum y value and the time it takes to return to $y=0$.

Solution. We introduce

$$\bar{y} = \frac{y}{y_c}, \quad \bar{t} = \frac{t}{t_c}\,.$$

Inserted in the formula we get

$$y_c\bar{y} = v_0 t_c \bar{t} - \frac{1}{2} g t_c^2 \bar{t}^2 \,.$$

1. At the maximum point of y, $y' = 0$, so $y' = v_0 - gt = 0$, which means $t = v_0/g$ and $y_{\max} = v_0 v0/g - \frac{1}{2} g v_0^2/g^2 = \frac{1}{2} v_0^2/g$. We choose $t_c = v_0/g$ and $y_c = \frac{1}{2} v_0^2/g$. This gives

$$\frac{1}{2}\frac{v_0^2}{g}\bar{y} = \frac{v_0^2}{g}\bar{t} - \frac{1}{2}\frac{v_0^2}{g}\bar{t}^2 \quad \Rightarrow \quad \bar{y} = 2\bar{t} - \bar{t}^2 \,.$$

2. The body is back at $y = 0$ for $v_0 t - \frac{1}{2} g t^2 = 0$, which gives $t_c = 2v_0/g$ and $y_c = 2y_{\max} = v_0^2/g$. Inserted, we get

$$\frac{v_0^2}{g}\bar{y} = 2\frac{v_0^2}{g}\bar{t} - \frac{1}{2} 4\frac{v_0^2}{g}\bar{t}^2 \quad \Rightarrow \quad \bar{y} = 2\bar{t}(1 - \bar{t}) \,.$$

Observe that the physical parameters v_0 and g are absent in the scaled formula.
Filename: `vertical_motion`.

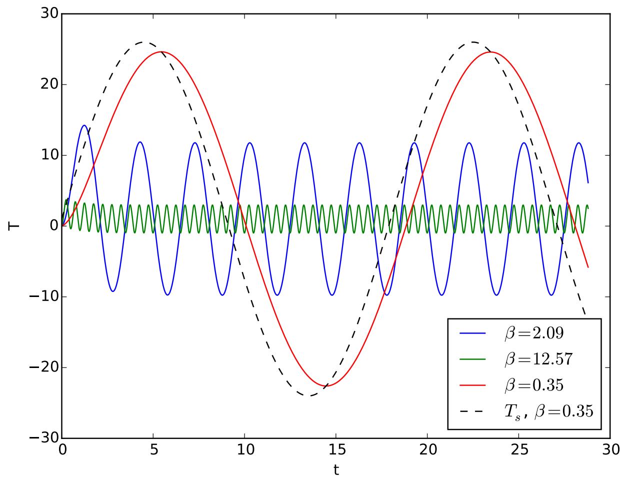
Exercise 2.3: Perform alternative scalings

The problem in Section 2.1.8 applies a temperature scaling

$$\bar{T} = \frac{T - T_0}{T_m - T_0},$$

which is not always suitable.

a) Consider the case $T_0 = T_m$ and the fact that $|T_m - T_0|$ does not represent the characteristic temperature scale since it collapses to zero. Formulate a suitable scaling in this case. The figure below corresponds to $T_m = 25$ C, $T_0 = 24.9$ C, and $a = 2.5$ C. We clearly see that $\bar{T}$ is not of size unity.



Solution. The typical temperature variations will now be oscillations of amplitude a around $T_m = T_0$, so $2a$ is the typical variation of the surrounding temperature. If the time scale of T_s is sufficiently large (or more precisely, β is small), the temperature will actually reach amplitudes of size a, but for fast oscillations in T_s, there will not be enough time to transfer heat to/from the body, so the amplitudes of T will be smaller. Taking $2a$ to be the typical temperature range, we can propose the scaling

$$\bar{T} = \frac{T - T_m}{2a}\,.$$

Inserted in the differential equation, we get with $t_c = 1/k$,

$$k2a\frac{\bar{T}}{d\bar{t}} = -k(2a\bar{T} + T_m - (T_m + a\sin(\omega t))),$$

which simplifies to

$$\frac{\bar{T}}{d\bar{t}} = -(\bar{T} - \frac{1}{2}\sin(\beta t)),$$

where $\beta = \omega/k$. The initial condition becomes

$$\bar{T}(0) = -\frac{1}{2}\alpha,$$

where $\alpha = a/(T_m - T_0)$ is the dimensionless number that appeared in the scaled differential equation in Section 2.1.8.

b) Consider the case where a is much larger than $|T_m - T_0|$. What is an appropriate scaling of the temperature?

Solution. In this case, T will oscillate around T_m and at maximum reach the amplitude a if β is small, see the figure in a). This is the same situation as in a), and we can consequently use the same scaling and obtain the same scaled problem.

Problem 2.4: A nonlinear ODE for vertical motion with air resistance

The velocity $v(t)$ of a body moving vertically through a fluid in the gravity field, with fluid drag and buoyancy, is governed by the ODE

$$mv' = -\frac{1}{2}C_D\varrho A|v|v - mg + \varrho V g, \quad v(0) = v_0,$$

where t is time, m is the mass of the body, C_D is a drag coefficient, ϱ is the density of the fluid, A is the cross-sectional area perpendicular to the motion, g is the acceleration of gravity, and V is the volume of the body. Scale this ODE.

Solution. We introduce as usual

$$\bar{v} = \frac{v}{v_c}, \quad \bar{t} = \frac{t}{t_c},$$

but the main challenge is to find values for v_c and t_c. Inserting the scaled quantities gives

$$m\frac{v_c}{t_c}\frac{d\bar{v}}{d\bar{t}} = -\frac{1}{2}C_D\varrho A v_c^2|\bar{v}|\bar{v} - mg + \varrho V g, \quad v_c v(0) = v_0,$$

It is tempting to set $v_c = v_0$, but $v_0 = 0$ is a relevant value so this choice is not good. The motion is of decay type so t_c and v_c should be based on characteristics of the decay. The terminal velocity, defined by $v' = 0$, is

$$v_T = \sqrt{\frac{2(\varrho V - m)g}{C_D\varrho A}},$$

when $\varrho V > m$ such that the buoyancy wins over gravity and the motion is upwards. Otherwise,

$$v_T = -\sqrt{\frac{2(m - \varrho V)g}{C_D\varrho A}}\,.$$

The two formulas can be combined to

$$v_T = \mathrm{sign}(\varrho V - m)\sqrt{\frac{2|\varrho V - m|g}{C_D\varrho A}}\,.$$

We take $v_c = |v_T| = \sqrt{\frac{2|\varrho V - m|g}{C_D \varrho A}}$. This results in

$$\frac{d\bar{v}}{d\bar{t}} = -t_c \frac{1}{2m} C_D \varrho A \sqrt{\frac{2|\varrho V - m|g}{C_D \varrho A}} |\bar{v}|\bar{v} - t_c \sqrt{\frac{C_D \varrho g A}{2|\varrho V - m|}} (1 - \frac{\varrho V}{m}),$$

and

$$v(0) = v_0 \sqrt{\frac{C_D \varrho A}{2g|\varrho V - m|}}\,.$$

A natural choice is to assume $d\bar{v}/d\bar{t}$ and $\bar{v}$ to be of the same order, which means that coefficient in front of the nonlinear term $|\bar{v}|\bar{v}$ should be unity. This forces t_c to be

$$t_c = \frac{2m}{\sqrt{2g|\varrho V - m|C_D \varrho A}}\,.$$

Introducing the dimensionless numbers

$$\alpha = \frac{\varrho V}{m}, \quad \beta = v_0 \sqrt{\frac{C_D \varrho A}{2g|\varrho V - m|}} = \frac{v_0}{|v_T|},$$

we get the scaled ODE problem

$$\frac{d\bar{v}}{d\bar{t}} = -|\bar{v}|\bar{v} + \text{sign}(1 - \alpha), \quad \bar{v}(0) = \beta\,.$$

Note that, as usual, the dimensionless numbers have simple interpretations: α is the ratio of the mass of the displaced fluid and the mass of the body, while β is the ratio of the initial and terminal velocities.
Filename: `vertical_motion_with_drag`.

Exercise 2.5: Solve a decay ODE with discontinuous coefficient

Make software for the problem in Section 2.1.6 so that you can produce Figure 2.3.

Hint. Follow the ideas for software in Section 2.1.5: use the `decay_vc.py` module as computational engine and modify the `falling_body.py` code.

Solution. We use `joblib` to avoid unnecessary execution of the scaled problem, as explained in Section 2.1.4. A potential complete program is listed below.

```
import sys, os
# Enable loading modules in ../src-scaling
```

```
sys.path.insert(0, os.path.join(os.pardir, 'src-scaling'))
from decay_vc import solver as solver_unscaled
from math import pi
import matplotlib.pyplot as plt
import numpy as np

def solver_scaled(gamma, T, dt, theta=0.5):
    """
    Solve u'=-a*u, u(0)=1 for (0,T] with step dt and theta method.
    a=1 for t < gamma and 2 for t > gamma.
    """
    print 'Computing the numerical solution'
    return solver_unscaled(
        I=1, a=lambda t: 1 if t < gamma else 5,
        b=lambda t: 0, T=T, dt=dt, theta=theta)

import joblib
disk_memory = joblib.Memory(cachedir='temp')
solver_scaled = disk_memory.cache(solver_scaled)

def unscale(u_scaled, t_scaled, d, I):
    return I*u_scaled, d*t_scaled

def main(d,
         I,
         t_1,
         dt=0.04, # Time step, scaled problem
         T=4,     # Final time, scaled problem
         ):

    legends1 = []
    legends2 = []
    plt.figure(1)
    plt.figure(2)

    gamma = t_1*d
    print 'gamma=%.3f' % gamma
    u_scaled, t_scaled = solver_scaled(gamma, T, dt)

    plt.figure(1)
    plt.plot(t_scaled, u_scaled)
    legends1.append('gamma=%.3f' % gamma)

    plt.figure(2)
    u, t = unscale(u_scaled, t_scaled, d, I)
    plt.plot(t, u)
    legends2.append('d=%.2f [1/s], t_1=%.2f s' % (d, t_1))
    plt.figure(1)
    plt.xlabel('scaled time'); plt.ylabel('scaled velocity')
    plt.legend(legends1, loc='upper right')
    plt.savefig('tmp1.png');  plt.savefig('tmp1.pdf')

    plt.figure(2)
    plt.xlabel('t [s]');  plt.ylabel('u')
```

```
    plt.legend(legends2, loc='upper right')
    plt.savefig('tmp2.png');  plt.savefig('tmp2.pdf')
    plt.show()

if __name__ == '__main__':
    main(d=1/120., I=1, t_1=100)
```

Filename: `decay_jump`.

Exercise 2.6: Implement a scaled model for cooling

Use software for the unscaled problem (2.16) to compute the solution of the scaled problem (2.23). Let T_s be a function of time.

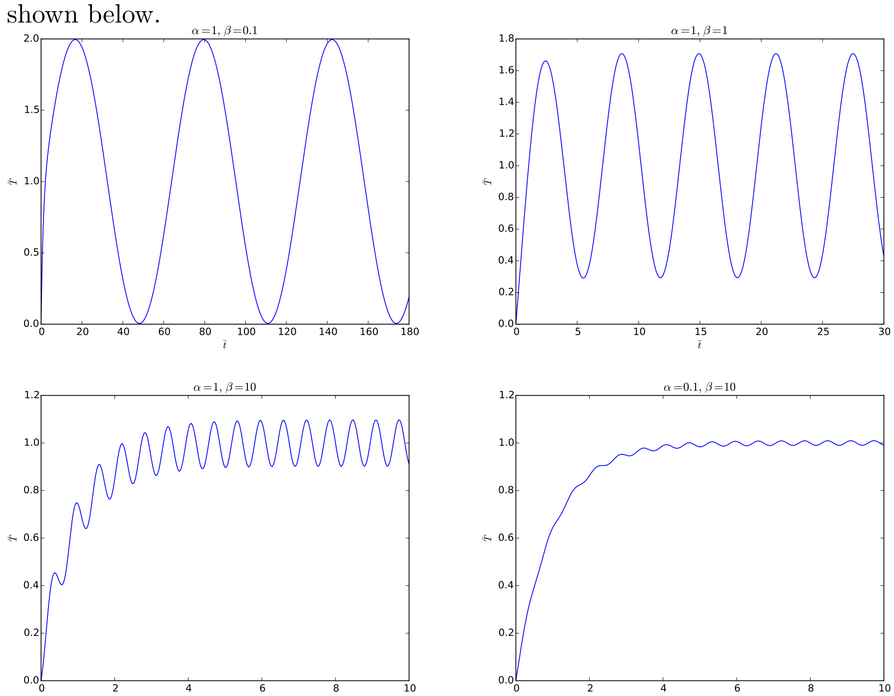
Hint. You may use the general software `decay_vc.py` for computing with the cooling model. See Section 2.1.5 for more ideas.

Solution. The problem (2.16) is just a special case of the general problem $u' = -au + b$ solved by the `decay_vc` module. We can make an implementation of (2.23) in terms of the model $u' = -au + b$:

```
import sys, os
# Enable loading modules in ../src-scaling
sys.path.insert(0, os.path.join(os.pardir, 'src-scaling'))
from decay_vc import solver as solver_unscaled
from math import pi
import matplotlib.pyplot as plt
import numpy as np

def solver_scaled(alpha, beta, t_stop, dt, theta=0.5):
    """
    Solve T' = -T + 1 + alha*sin(beta*t), T(0)=0
    for (0,T] with step dt and theta method.
    """
    print 'Computing the numerical solution'
    return solver_unscaled(
        I=0, a=lambda t: 1,
        b=lambda t: 1 + alpha*np.sin(beta*t),
        T=t_stop, dt=dt, theta=theta)

import joblib
disk_memory = joblib.Memory(cachedir='temp')
solver_scaled = disk_memory.cache(solver_scaled)

def main(alpha,
         beta,
         t_stop=50,
         dt=0.04
         ):

    T, t = solver_scaled(alpha, beta, t_stop, dt)
```

```
    plt.plot(t, T)
    plt.xlabel(r'$\bar t$');  plt.ylabel(r'$\bar T$')
    plt.title(r'$\alpha=%g,\ \beta=%g$' % (alpha, beta))
    filestem = 'tmp_%s_%s' % (alpha, beta)
    plt.savefig(filestem + '.png');  plt.savefig(filestem + '.pdf')
    plt.show()

if __name__ == '__main__':
    import sys
    alpha = float(sys.argv[1])
    beta = float(sys.argv[2])
    t_stop = float(sys.argv[3])
    main(alpha, beta, t_stop)
```

Simulations for $\alpha = 1$ and $\beta = 0.1, 1, 10$ as well as for $\alpha = 0.1$ and $\beta = 1$ are shown below.



Filename: `cooling1`.

Problem 2.7: Decay ODE with discontinuous coefficients

The goal of this exercise is to scale the problem $u'(t) = -a(t)u(t) + b(t)$, $u(0) = I$, when

$$a(t) = \begin{cases} Q, & t < s, \\ Q - A, & t \geq s, \end{cases} \qquad b = \begin{cases} \epsilon t, & t < s, \\ 0, & t \geq s, \end{cases}$$

Here, $Q, A, \epsilon > 0$.

Solution. We start by scaling the known functions a and b. Since $Q, A.\epsilon > 0$, $\max|a(t)| = Q$ and $\max|b(t)| = \epsilon s$. Scaled versions of these functions are then

$$\bar{a} = \frac{a}{Q}, \quad \bar{b} = \frac{b}{\epsilon s}.$$

As usual, we scale u and t as

$$\bar{u} = \frac{u}{u_c}, \quad \bar{t} = \frac{t}{t_c}.$$

The scaled ODE reads

$$\frac{d\bar{u}}{d\bar{t}} = -t_c Q\bar{a}(\bar{t}) + u_c^{-1} t_c \epsilon s \bar{b}.$$

A natural choice of u_c is $u_c = I$. The a term will reduce $\bar{u}$ from 1, while the b term may have a growth effect.

The time scale is best chosen to reflect the dynamics of the process, i.e., the decay with strength Q, so we set $t_c = 1/Q$. This choice results in

$$\frac{d\bar{u}}{d\bar{t}} = -\bar{a}(\bar{t}) + \alpha\bar{b},$$

with

$$\bar{a}(\bar{t}) = \begin{cases} 1, & \bar{t} < \gamma, \\ 1-\beta, & \bar{t} \geq \gamma, \end{cases}$$

and

$$\bar{b}(\bar{t}) = \begin{cases} \gamma^{-1}\bar{t}, & \bar{t} < \gamma, \\ 0, & \bar{t} \geq \gamma, \end{cases}$$

The initial condition is $\bar{u}(0) = 1$. We have three dimensionless numbers in the problem:

$$\alpha = \frac{\epsilon s}{QI}, \quad \beta = \frac{A}{Q}, \quad \gamma = Qs.$$

We realize that α measures the ratio of the b term (ϵs) and the au term (QI), β reflects the relative jump in a, while γ measures the ratio of the transition point $t = s$ and the characteristic time scale.
Filename: `decay_varcoeff`.

Exercise 2.8: Alternative scalings of a cooling model

Implement the scaled model (2.29) and produce a plot with curves corresponding to various values of α and p to summarize how $\bar{u}(\bar{t})$ looks like.

Hint. A centered Crank-Nicolson-style scheme for (2.29) can use an old time value for the nonlinear coefficient:

$$\frac{\bar{u}^{n+1}-\bar{u}^n}{\Delta t} = (1-\alpha\bar{u}^n)^p\frac{1}{2}(\bar{u}^n+\bar{u}^{n+1})\,.$$

Filename: `growth`.

Exercise 2.9: Projectile motion

We have the following mathematical model for the motion of a projectile in two dimensions:

$$m\ddot{\boldsymbol{x}} + \frac{1}{2}C_D\varrho A|\dot{\boldsymbol{x}}|\dot{\boldsymbol{x}} = -mg\boldsymbol{j},\quad \boldsymbol{x}(0)=\boldsymbol{0},\ \dot{\boldsymbol{x}}(0)=v_0\cos\theta\boldsymbol{i}+v_0\sin\theta\boldsymbol{j}\,.$$

Here, m is the mass of the projectile, $\boldsymbol{x}=x\boldsymbol{i}+y\boldsymbol{j}$ is the position vector of the projectile, $\boldsymbol{i}$ and $\boldsymbol{j}$ are unit vectors along the x and y axes, respectively, $\ddot{\boldsymbol{x}}$ and $\dot{\boldsymbol{x}}$ is the second- and first-order time derivative of $\boldsymbol{x}(t)$, C_D is a drag coefficient depending on the shape of the projectile (can be taken as 0.4 for a sphere), ϱ is the density of the air, A is the cross section area (can be taken as πR^2 for a sphere of radius R), g is gravity, v_0 is the initial velocity of the projectile in a direction that makes the angle θ with the ground.

a) Neglect the air resistance term proportional to $\dot{\boldsymbol{x}}$ and solve analytically for $\boldsymbol{x}(t)$.

Solution. The vector differential equation reduces to the two component equations

$$m\ddot{x}(t)=0,\quad m\ddot{y}(t)=-mg\,.$$

Integrating twice yields

$$x(t)=C_1t+C_2,\quad y(t)=-\frac{1}{2}gt^2+C_3t+C_4\,.$$

The condition $\boldsymbol{x}(0)=\boldsymbol{0}$ forces $C_2=C_4=0$. The condition on the derivative gives $C_1=v_0\cos\theta$ and $C_3=v_0\sin\theta$. The result is therefore

$$\boldsymbol{x}(t)=v_0\cos(\theta)t\boldsymbol{i}+(v_0\sin(\theta)t-\frac{1}{2}gt^2)\boldsymbol{j}\,.$$

b) Make the model for projectile motion with air resistance non-dimensional. Use the maximum height from the simplification in a) as length scale.

Solution. We introduce dimensionless quantities:

$$\bar{x} = \frac{x}{L}, \quad \bar{y} = \frac{y}{L}, \quad \bar{t} = \frac{t}{t_c},$$

where the scales L and t_c must be determined. Inserted in the original equation:

$$\frac{mL}{t_c^2}\frac{d^2\bar{\boldsymbol{x}}}{d\bar{t}^2} + \frac{1}{2}C_D\varrho A\frac{L^2}{t_c^2}\left|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}}\right|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}} = -mg\boldsymbol{j}\,.$$

Dividing by mL/t_c^2 gives

$$\frac{d^2\bar{\boldsymbol{x}}}{d\bar{t}^2} + \frac{1}{2}C_D\varrho A\frac{L}{m}\left|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}}\right|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}} = -\frac{gt_c^2}{L}\boldsymbol{j}\,.$$

It is tempting to determine scales from setting coefficients in this equation to unity. However, we expect the effect of air resistance to be (much) smaller than gravity, so the primary balance in the equation is between the acceleration term and the gravity term. Setting the coefficient in the gravity term to unity gives $L = gt_c^2$, which provides a relevant length scale. However, setting the coefficient in the air resistance term to unity gives a length scale only relevant when air resistance is as important as gravity and acceleration, and that might be the case for a very hard kick of a soccer ball, for instance. Otherwise, the coefficient in front of the air resistance term will be a dimensionless number which is expected to be small. Of these reasons, we need to determine L from the insight in a) when we solved the problem using a balance of acceleration and gravity only.

The maximum height $y_{\max}$ occurs when $\dot{y} = 0$, and from the solution in a) we get

$$\dot{y} = v_0\sin\theta - gt = 0 \quad \Rightarrow \quad t = g^{-1}v_0\sin\theta\,.$$

The corresponding $y_{\max}$ value is

$$y_{\max} = g^{-1}v_0^2\sin^2\theta - \frac{1}{2}g^{-1}v_0^2\sin^2\theta = \frac{1}{2}g^{-1}v_0^2\sin^2\theta\,.$$

We can take $L = y_{\max}$ and let t_c be the corresponding t value: $t_c = g^{-1}v_0\sin\theta$. Inserted in the scaled problem:

$$\frac{d^2\bar{\boldsymbol{x}}}{d\bar{t}^2} + \frac{1}{2}C_D\varrho A\frac{v_0^2\sin^2\theta}{2mg}\left|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}}\right|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}} = -2\boldsymbol{j}\,.$$

We can identify a dimensionless parameter

$$\alpha = \frac{C_D\varrho A v_0^2\sin^2\theta}{4mg},$$

and write the scaled equation as

$$\frac{d^2\bar{\boldsymbol{x}}}{d\bar{t}^2}+\alpha\left|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}}\right|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}}=-2\boldsymbol{j},$$

with initial conditions

$$\bar{\boldsymbol{x}}(0)=\mathbf{0},\quad \frac{d\bar{\boldsymbol{x}}}{d\bar{t}}(0)=\frac{t_c}{L}(v_0\cos\theta\boldsymbol{i}+v_0\sin\theta\boldsymbol{j})=2\cot\theta\boldsymbol{i}+2\boldsymbol{j}\,.$$

Apart from the factor 4, the α formula as stated above is seen to reflect the air resistance force in the vertical motion (which has velocity $v_0\sin\theta$) and the gravity force. We can compute α for a soft and a hard kick of a soccer ball as described in d), and the values are 0.04 and 0.2, respectively, showing that the balance of acceleration and gravity is relevant - even a hard kick only gives $\alpha 0.6$, but in that regime it would not be wrong to choose L such that the coefficient in the air resistance term becomes unity.

Basing t_c and L on the entire flight back to $y=0$ means $t_c=2g^{-1}v_0\sin\theta$ and $L=2y_{\max}$ (the total vertical distance), removes the factor 2 on the right-hand side and reduces α by a factor 2.

c) Make the model dimensionless again, but this time by demanding that the scaled initial velocity is unity in x direction.

Solution. The scaled initial velocity condition is

$$\bar{x}(0)=\frac{t_c}{L}v_0\cos\theta\,.$$

Demanding the scaled velocity to be unity gives

$$L=t_cv_0\cos\theta\,.$$

The scaled initial velocity in y direction becomes

$$\bar{y}(0)=\tan\theta\,.$$

The scaled ODE becomes

$$\frac{d^2\bar{\boldsymbol{x}}}{d\bar{t}^2}+\frac{1}{2}C_D\varrho A\frac{L}{m}\left|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}}\right|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}}=-\frac{gt_c}{v_0\cos\theta}\boldsymbol{j}\,.$$

We can choose t_c such that the gravity term is unity and balances the acceleration term:

$$t_c=g^{-1}v_0\cos\theta,$$

which makes

$$L=t_cv_0\cos\theta=g^{-1}v_0^2\cos^2\theta\,.$$

The coefficient in the drag term becomes

$$\alpha = \frac{C_D \varrho A v_0^2 \cos^2\theta}{2mg}\,.$$

To summarize, we get the scaled problem

$$\frac{d^2\bar{\boldsymbol{x}}}{d\bar{t}^2} + \alpha\left|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}}\right|\frac{d\bar{\boldsymbol{x}}}{d\bar{t}} = -\boldsymbol{j}, \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{0}, \quad \frac{d\bar{\boldsymbol{x}}}{d\bar{t}}(0) = \boldsymbol{i} + \tan\theta\boldsymbol{j}\,.$$

d) A soccer ball has radius 11 cm and mass 0.43 kg, the density of air is 1.2 kgm^{-3}, a soft kick has velocity 30 km/h, while a hard kick may have 120 km/h. Estimate the dimensionless parameter in the scaled problem for a soft and a hard kick with θ corresponding to 45 degrees. Solve the scaled differential equation for these values and plot the trajectory (y versus x) for the two cases.

Solution. We need to express R, v_0, and θ in standard SI units: $A = \pi 0.11^2$ m^2, $\theta = 45\cdot\pi/180$ rad, $v_0 = 30/3.6$ and $120/3.6$ m/s. The formula for α results in

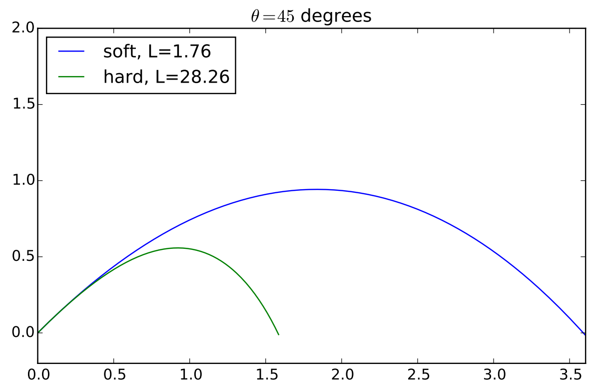
$$\alpha_{\mathrm{soft}} \approx 0.037, \quad \alpha_{\mathrm{hard}} \approx 0.6\,.$$

Appropriate computer code appears below (using Odespy to solve the ODE system as a first-order system).

```
import matplotlib.pyplot as plt
import odespy
import numpy as np

def solver(alpha, ic, T, dt=0.05):
    def f(u, t):
        x, vx, y, vy = u
        v = np.sqrt(vx**2 + vy**2)  # magnitude of velocity
        system = [
            vx,
            -alpha*np.abs(v)*vx,
            vy,
            -2 - alpha*np.abs(v)*vy,
            ]
        return system

    Nt = int(round(T/dt))
    t_mesh = np.linspace(0, Nt*dt, Nt+1)

    solver = odespy.RK4(f)
    solver.set_initial_condition(ic)
    u, t = solver.solve(t_mesh,
                        terminate=lambda u, t, n: u[n][2] < 0)
    x = u[:,0]
    y= u[:,2]
    return x, y, t

def demo_soccer_ball():
```

```
    import math
    theta_degrees = 45
    theta = math.radians(theta_degrees)
    ic = [0, 2/math.tan(theta), 0, 2]
    g = 9.81
    v0_s = 8.3    # soft kick
    v0_h = 33.3   # hard kick
    # Length scales
    L_s = 0.5*(v0_s**2/g)*math.sin(theta)**2
    L_h = 0.5*(v0_h**2/g)*math.sin(theta)**2
    print 'L:', L_s, L_h

    m = 0.43  # kg
    R = 0.11  # m
    A = math.pi*R**2
    rho = 1.2 # kg/m^3
    C_D = 0.4
    alpha_s = C_D*rho*A*v0_s**2*math.cos(theta)**2/(4*m*g)
    alpha_h = C_D*rho*A*v0_h**2*math.cos(theta)**2/(4*m*g)
    print 'alpha:', alpha_s, alpha_h
    x_s, y_s, t = solver(alpha=alpha_s, ic=ic, T=6, dt=0.01)
    x_h, y_h, t = solver(alpha=alpha_h, ic=ic, T=6, dt=0.01)
    plt.plot(x_s, y_s, x_h, y_h)
    plt.legend(['soft, L=%.2f' % L_s, 'hard, L=%.2f' % L_h],
               loc='upper left')
    # Let the y range be [-0.2,2] so we have space for legends
    plt.axis([x_s[0], x_s[-1], -0.2, 2])
    plt.axes().set_aspect('equal') # x and y axis have same scaling
    plt.title(r'$\theta=%d$ degrees' % theta_degrees)
    plt.savefig('tmp.png')
    plt.savefig('tmp.pdf')
    plt.show()

demo_soccer_ball()
```

For $\theta = 45$ degrees we get the plot



The blue curve is very close to motion without air resistance. We clearly see how significant air resistance is once the velocity is large enough. The total length is approximately 6.3 m for a soft kick and 45 m for a hard kick (multiply the dimensionless lengths in the plot by the corresponding L).
Filename: `projectile`.

Problem 2.10: A predator-prey model

The evolution of animal populations with a predator and a prey (e.g., lynx and hares, or foxes and rabbits) can be described by the Lotka-Volterra ODE system

$$H' = H(a - bL), \tag{2.93}$$

$$L' = L(dH - c), \tag{2.94}$$

$$H(0) = H_0, \tag{2.95}$$

$$L(0) = L_0\,. \tag{2.96}$$

Here, H is the number of animals of the prey (say hares) and L is the corresponding measure of the predator population (say lynx). There are six parameters: a, b, c, d, H_0, and L_0.

The terms have the following meanings:

- aH is the exponential population growth of H due to births and deaths and is governed by the access to nutrition,
- $-bHL$ is the loss of preys because they are eaten by predators,

- dHL is the increase of predators because they eat preys (but only a fraction of the eaten preys, bHL, contribute to population growth of the predator and therefore $d < b$),
- $-cL$ is the exponential decay in the predator population because of deaths (the increase is modeled by dHL).

Dimensionless independent and dependent variables are introduced as usual by

$$\bar{t} = \frac{t}{t_c}, \quad \bar{H} = \frac{H}{H_c}, \quad \bar{L} = \frac{L}{L_c},$$

where t_c, H_c, and L_c are scales to be determined. Inserted in the ODE problem we arrive at

$$\frac{H_0}{t_c}\frac{d\bar{H}}{d\bar{t}} = H_0\bar{H}(a - bH_0\bar{L}), \tag{2.97}$$

$$\frac{H_0}{t_c}\frac{d\bar{L}}{d\bar{t}} = H_0\bar{L}(dH_0\bar{H} - c), \tag{2.98}$$

$$H_c\bar{H}(0) = H_0, \tag{2.99}$$

$$L_c\bar{H}(0) = L_0\,. \tag{2.100}$$

a) Consider first a simple, intuitive scaling of H and L based on initial conditions $H_c = H_0$ and $L_c = H_c$. This means that $\bar{H}$ starts out at unity and $\bar{L}$ starts out as the fraction L_0/H_0. Find a time scale and identify dimensionless parameters in the scaled ODE problem.

Solution. With $H_c = L_c = H_0$ in (2.97)-(2.100) we get

$$\begin{aligned}
\frac{d\bar{H}}{d\bar{t}} &= \frac{a}{bH_0}\bar{H} - \bar{L}\bar{H},\\
\frac{d\bar{L}}{d\bar{t}} &= \frac{d}{b}\bar{L}\bar{H} - \frac{c}{bH_0}\bar{L}),\\
\bar{H}(0) &= 1,\\
\bar{L}(0) &= \frac{L_0}{H_0}\,.
\end{aligned}$$

With the dimensionless parameters

$$\alpha = \frac{a}{bH_0}, \quad \beta = \frac{d}{b}, \quad \gamma = \frac{c}{bH_0}, \quad \delta = \frac{H_0}{L_0},$$

we can write the dimensionless problem as

$$
\begin{aligned}
\frac{d\bar{H}}{d\bar{t}} &= \alpha\bar{H} - \bar{L}\bar{H},\\
\frac{d\bar{L}}{d\bar{t}} &= \beta\bar{L}\bar{H} - \gamma\bar{L}),\\
\bar{H}(0) &= 1,\\
\bar{L}(0) &= \delta\,.
\end{aligned}
$$

The quantity bH_0 is the number of eaten preys per predator. Then α measures the ratio of natural population growth of the prey, due to nutrition, and the number of eaten preys per predator. The β parameter measures the fraction of the eaten preys and the amount of this that actually leads to population growth of the predator. The number γ reflects the ratio of predator deaths and the eaten preys per predator, and δ is the initial fraction of preys and predators.

b) Try a different scaling where the aim is to adjust the scales such that the ODEs become as simple as possible, i.e, have as few dimensionless parameters as possible. Compare with the scaling in a).

Solution. Dividing by H_c and L_c in (2.97) and (2.100), respectively, and multiply by t_c:

$$
\begin{aligned}
\frac{d\bar{H}}{d\bar{t}} &= t_c\bar{H}(a - bL_c\bar{L}),\\
\frac{d\bar{L}}{d\bar{t}} &= t_c\bar{L}(dH_c\bar{H} - c)\,.
\end{aligned}
$$

Choosing $t_c = 1/a$ and $t_c a L_c = 1$, i.e., $L_c = a/b$, makes the first equation free of parameters: $\bar{H}' = \bar{H}(1-\bar{L})$. Factoring out c in the equation for L and choosing $H_c d/c = 1$, i.e., $H_c = c/d$, leaves us with the L equation as $\bar{L}' = (c/a)\bar{L}(\bar{H}-1)$. The ratio c/a is now called μ and equals γ/α from a).

The initial conditions lead to $\bar{H}(0) = H_0/H_c = H_0 d/c = \beta/\gamma = \nu$, and $\bar{L}(0) = L_0/L_c = L_0 b/a = \delta/\alpha = \omega$.

The dimensionless problem is now

$$
\begin{aligned}
\frac{d\bar{H}}{d\bar{t}} &= \bar{H}(1-\bar{L}), && (2.101)\\
\frac{d\bar{L}}{d\bar{t}} &= \mu\bar{L}(\bar{H}-1) = \gamma\alpha^{-1}\bar{L}(\bar{H}-1), && (2.102)\\
\bar{H}(0) &= \nu = \beta/\gamma, && (2.103)\\
\bar{L}(0) &= \omega = \delta/\alpha, && (2.104)
\end{aligned}
$$

with

$$\mu = \frac{c}{a}, \quad \nu = H_0\frac{d}{c}, \quad \omega = L_0\frac{b}{a}\,.$$

The unknowns $\bar{H}$ and $\bar{L}$ now has less intuitive scalings,

$$\bar{H} = \frac{Hd}{c}, \quad \bar{L} = \frac{Lb}{a},$$

while time is measured in the units based on the exponential growth due to births and deaths of preys (a). The number of dimensionless parameters is one less since we have one more scale (for L_c) at our disposal. Simplicity in one initial conditions in a) is exchanged with more simplicity in the ODEs, which now have only one dimensionless parameter.

Note that ν and ω must be different from unity to avoid $\bar{H} \neq 0$ and $\bar{L} \neq 0$ because of the factors $1-L$ and $H-1$ in the equations that can make $\bar{H}' = 0$ and $\bar{L}' = 0$.

c) A more mathematical approach to determining suitable scales for H and L consists in finding the stationary points (H,L) of the ODE system, where $H' = L' = 0$, and use such points as characteristic sizes of the dependent variables. Show that $H' = L' = 0$ implies $H = L = 0$ or $L = a/b$ and $H = c/d$. Use $H_c = a/b$, $L_c = c/d$, and find a time scale. Compare with the result in b).

Solution. Setting $H' = L' = 0$ leads to

$$H(a - bL) = 0, \quad L(dH - c) = 0,$$

from which we see that the factors must vanish: $H = L = 0$, $L = a/b$, and $H = c/d$. With Use $H_c = a/b$, $L_c = c/d$, and $t_c = 1/a$ we get the same scaling as in b), but with a different motivation.
Filename: `predator_prey`.

Problem 2.11: A model for competing species

Let $N_1(t)$ and $N_2(t)$ be the number of animals in two competing species. A generalized Lotka-Volterra model is based on a logistic growth of each specie and a predator-prey like interaction (cf. Problem 2.10):

$$\frac{dN_1}{dt} = r_1N_1\left(1 - \frac{N_1}{M_1} - s_{12}\frac{N_2}{M_1}\right), \tag{2.105}$$

$$\frac{dN_2}{dt} = r_2N_2\left(1 - \frac{N_2}{M_2} - s_{21}\frac{N_1}{M_2}\right), \tag{2.106}$$

where r_1, r_2, M_1, M_2, s_{12}, and s_{21} are given constants. The initial conditions specify N_1 and N_2 at $t = 0$. Find suitable scales and derive a dimensionless ODE problem.

Solution. As always, we can introduce dimensionless variables, We use ideas from scaling of ODEs for logistic growth, i.e., we use the carrying capacities M_1 and M_2 as characteristic (maximum) values of N_1 and N_2, respectively. Time can be scaled from the initial exponential growth of N_1 or N_2, i.e., $t_c = 1/r_1$ or $t_c = 1/r_2$. We choose the former here. Introducing

$$\bar{t} = r_1 t, \quad u_1 = \frac{N_1}{M_1}, \quad u_2 = \frac{N_2}{M_2},$$

in the ODE system, leads to

$$\begin{aligned}
\frac{du_1}{d\bar{t}} &= u_1\left(1 - u_1 - b_{21}\beta\gamma\right),\\
\frac{du_2}{d\bar{t}} &= \alpha u_2\left(1 - u_2 - b_{12}\beta^{-1}\gamma^{-1}\right),
\end{aligned}$$

where the dimensionless numbers are given by

$$\alpha = \frac{r_2}{r_1}, \quad \beta = \frac{M_2}{M_1}, \quad \gamma = \frac{b_{12}}{b_{21}}\,.$$

We have introduced two separate numbers β and γ since they are related to different parameters, but only their product matters. Alternatively, we could introduce the numbers $\mu = b_{12}\gamma$ and $\nu = b_{21}\gamma^{-1}$ in the last term of the first and second ODE, respectively.
Filename: `competing_species`.

Problem 2.12: Find the period of sinusoidal signals

This exercise aims at investigating various fundamental concepts like period, wave length, and frequency in non-damped and damped sinusoidal signals.

a) Plot the function

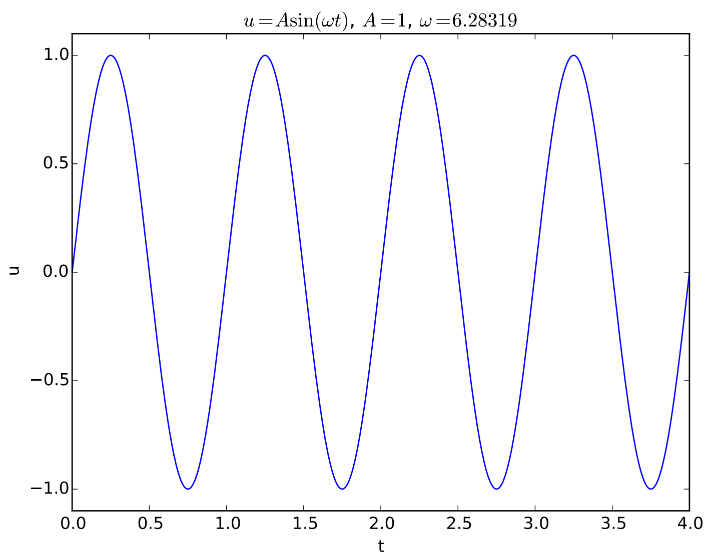
$$u(t) = A\sin(\omega t),$$

for $t \in [0, 8\pi/\omega]$. Choose ω and A.

Solution. Appropriate code is

```
import numpy as np
import matplotlib.pyplot as plt
```

```
def u(t, A, w, module=np):
    return A*module.sin(w*t)

def a():
    """Plot u."""
    w = 2*np.pi
    A = 1.0
    t = np.linspace(0, 8*np.pi/w, 1001)
    plt.figure()
    plt.plot(t, u(t, A, w))
    plt.xlabel('t');  plt.ylabel('u')
    plt.axis([t[0], t[-1], -1.1, 1.1])
    plt.title(r'$u=A\sin (\omega t)$, $A=%g$, $\omega = %g$'
              % (A, w))
    plt.savefig('tmp1.png');  plt.savefig('tmp1.pdf')
```



b) The *period* P of u is the shortest distance between two peaks (where $u = A$). Show mathematically that

$$P = \frac{2\pi}{\omega}\,.$$

Frequently, P is also referred to as the *wave length* of u.

Solution. Since the sine function has period 2π, we have that

$$\sin(\omega t) = \sin(\omega t + 2\pi)\,.$$

The definition of P is that sine gets its value again after time P:

$$\sin(\omega t) = \sin(\omega(t+P)).$$

Combing we get that $\sin(\omega t + 2\pi) = \sin(\omega(t+P))$, so the arguments must be equal:

$$\omega t + 2\pi = \omega(t+P),$$

from which it follows that $P = 2\pi/\omega$.

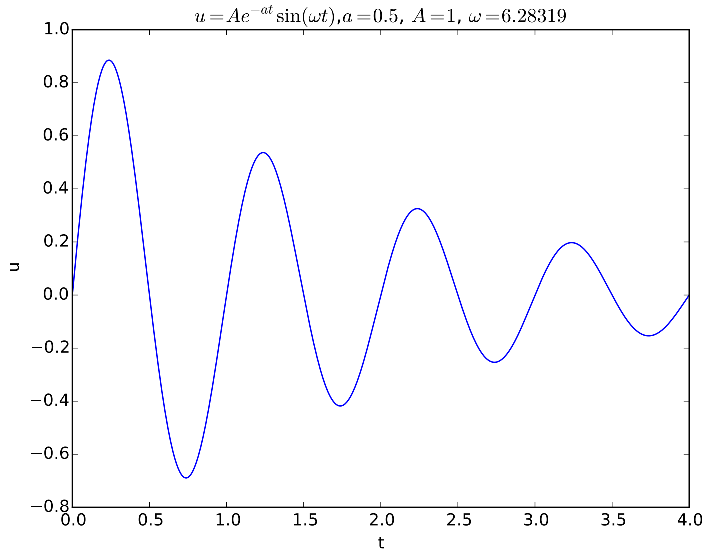
An alternative is to find the peaks as the points where $du/dt = 0$. Since $du/dt = \omega\cos(\omega t)$, this function is zero when $\omega t = n\pi$ for integer n. If $n\pi$ corresponds to a maximum, $(n+1)\pi$ will correspond to a minimum and $(n+2)\pi$ to the next maximum. The period P is the distance in time between two maxima:

$$\omega(t+P) - \omega t = (n + 2\pi - n\pi \quad \Rightarrow \quad P = \frac{2\pi}{\omega}.$$

c) Plot the damped signal $u(t) = e^{-at}\sin(\omega t)$ over four periods of $sin(\omega t)$. Choose ω, A, and a.

Solution. Code:

```
def u_damped(t, A, w, a, module=np):
    return A*module.exp(-a*t)*module.sin(w*t)

def c():
    """Plot damped u."""
    w = 2*np.pi
    A = 1.0
    a = 0.5
    t = np.linspace(0, 8*np.pi/w, 100001)
    plt.figure()
    plt.plot(t, u_damped(t, A, w, a))
    plt.xlabel('t');  plt.ylabel('u')
    plt.title(r'$u=Ae^{-at}\sin (\omega t)$,'
              '$a=%g$, $A=%g$, $\omega = %g$' % (a, A, w))
    plt.savefig('tmp2.png');  plt.savefig('tmp2.pdf')

    u_max = []
    u_ = u_damped(t, A, w, a)
    for i in range(1, len(t)-1):
        if u_[i-1] < u_[i] > u_[i+1]:
            u_max.append((t[i], u_[i]))
    print u_max
    for i in range(len(u_max)-1):
        print 'P=', u_max[i+1][0] - u_max[i][0]
```



d) What is the period of $u(t) = e^{-at}\sin(\omega t)$? We define the period P as the shortest distance between two peaks of the signal.

Hint. Use that $v = p\cos(\omega t) + q\sin(\omega t)$ can be rewritten as $v = B\cos(\omega t - \phi)$ with $B = \sqrt{p^2 + q^2}$ and $\phi = \tan^{-1}(p/q)$. Use such a rewrite of u' to find the peaks of u and then the period.

Solution. Finding the extrema from $u' = 0$ leads to

$$u' = -ae^{-at}\sin(\omega t) + e^{-at}\omega\cos(\omega t) = 0\,.$$

Using the hint to rewrite ($p = \omega$, $q = -a$), we get

$$u' = e^{-at}B\cos(\omega t - \phi) = 0, \quad B = \sqrt{\omega^2 + a^2},\ \phi = \tan^{-1}(-\omega/a)\,.$$

Now, e^{-at} is always positive so only the cosine function can cross zero, and that happens when the argument is $n\pi$ for integer n. However, all the maxima only occurs for $2n\pi$ (n integer). Demanding the argument to be $2n\pi$ we get the distance between two nearby peaks as

$$\omega(t + P) - \phi - (\omega t - \phi) = 2(n+1)\pi - 2n\pi,$$

which leads to

$$\omega P = 2\pi \quad \Rightarrow \quad P = \frac{2\pi}{\omega}\,.$$

The period of the damped signal is the same; only ω can alter the period.
Filename: `sine_period`.

Remarks. The *frequency* is the number of up and down cycles in one unit time. Since there is one cycle in a period P, the frequency is $f = 1/P$, measured in Hz. The *angular frequency* ω is then $\omega = 2\pi/P = 2\pi f$.

Problem 2.13: Oscillating mass with sliding friction

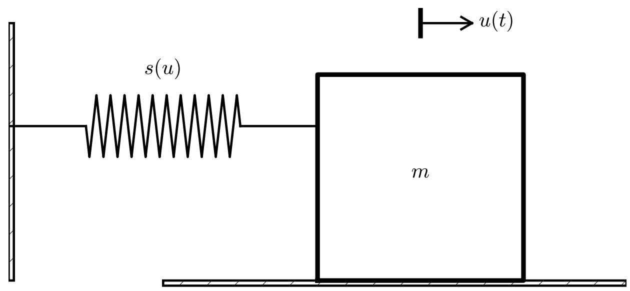


Fig. 2.13 Body sliding on a surface.

A mass attached to a spring is sliding on a surface and subject to a friction force, see Figure 2.13. The spring represents a force $-ku\boldsymbol{i}$, where k is the spring stiffness. The friction force is proportional to the normal force on the surface, $-mg\boldsymbol{j}$, and given by $-f(\dot{u})\boldsymbol{i}$, where

$$f(\dot{u}) = \begin{cases} -\mu mg, & \dot{u} < 0, \\ \mu mg, & \dot{u} > 0, \\ 0, & \dot{u} = 0 \end{cases}$$

Here, $\mu \geq 0$ is a friction coefficient. With the signum function

$$\mathrm{sign(x)} = \begin{cases} -1, & x < 0, \\ 1, & x > 0, \\ 0, & x = 0 \end{cases}$$

we can simply write $f(\dot{u}) = \mu mg\,\mathrm{sign}(\dot{u})$ (the sign function is implemented by `numpy.sign`).

The ODE problem for this one-dimensional oscillatory motion reads

$$m\ddot{u} + \mu mg\,\mathrm{sign}(\dot{u}) + ku = 0, \quad u(0) = I,\ \dot{u}(0) = V\,. \tag{2.107}$$

a) Scale the problem.

Solution. Inserting the dimensionless dependent and independent variables,

$$\bar{u} = \frac{u}{I}, \quad \bar{t} = \frac{t}{t_c},$$

in the problem gives

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \frac{t_c^2 \mu g}{I}\,\mathrm{sign}\left(\frac{d\bar{u}}{d\bar{t}}\right) + \frac{t_c^2 k}{m}\bar{u} = 0, \quad \bar{u}(0) = 1,\ \frac{d\bar{u}}{d\bar{t}}(0) = \frac{V t_c}{I}\,.$$

As usual, we base the characteristic time on the friction-free oscillations, which means a balance of the acceleration term and the spring term. That is, $t_c^2 k/m = 1$, and consequently $t_c = \sqrt{m/k}$.

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \frac{\mu m g}{kI}\,\mathrm{sign}\left(\frac{d\bar{u}}{d\bar{t}}\right) + \bar{u} = 0, \quad \bar{u}(0) = 1,\ \frac{d\bar{u}}{d\bar{t}}(0) = \frac{V\sqrt{mk}}{kI}\,.$$

Introducing the dimensionless variables

$$\alpha = \frac{\mu m g}{kI}, \quad \beta = \frac{V\sqrt{mk}}{kI},$$

the scaled problem can then be written

$$\frac{d^2\bar{u}}{d\bar{t}^2} + \alpha\,\mathrm{sign}\left(\frac{d\bar{u}}{d\bar{t}}\right) + \bar{u} = 0, \quad \bar{u}(0) = 1,\ \frac{d\bar{u}}{d\bar{t}}(0) = \beta\,.$$

The initial set of 6 parameters (μ, m, g, k, I, V) are reduced to 2 dimensionless combinations.

Let us check that the dimensionless parameters really are dimensionless. From the original ODE we know that each term has the dimension of force, i.e., $[\mathrm{MLT}^{-2}]$. Therefore, the friction coefficient μ is dimensionless since mg has dimension $[\mathrm{MLT}^{-2}]$, and k has dimension $[\mathrm{MT}^{-2}]$ since u has dimension [L]. Since I has the same dimension as u, kI has the dimension of $[\mathrm{MLT}^{-2}]$, which is the dimension of mg, and α is dimensionless. The β parameter has dimensions $[\mathrm{LT}^{-1}\mathrm{M}^{1/2}\mathrm{M}^{-1/2}\mathrm{TL}^{-1} = [1]$.

b) Implement the scaled model. Simulate for $\alpha = 0, 0.05, 0.1$ and $\beta = 0$.

Solution. We can use the package Odespy to solve the ODE. This requires rewriting the ODE as a system of two first-order ODEs:

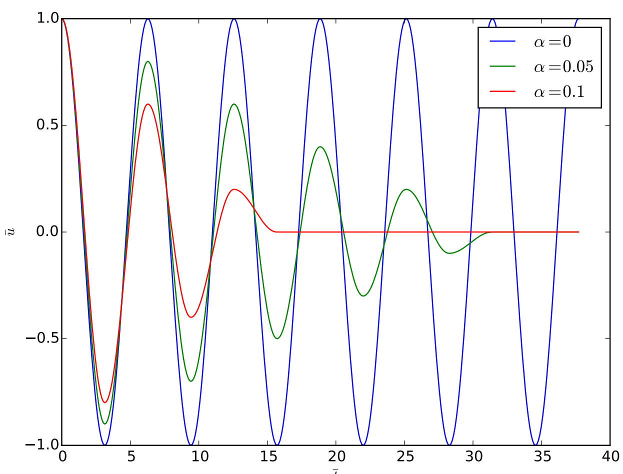
$$\begin{aligned} v' &= -\alpha\,\mathrm{sign}(v) - \bar{u}, \\ u' &= v, \end{aligned}$$

with initial conditions $v(0) = \beta$ and $u(0) = 1$. Here, $u(t)$ corresponds to the previous $\bar{u}(\bar{t})$, while $v(t)$ corresponds to $d\bar{u}/\bar{t}(\bar{t})$. Appropriate code is

```
import matplotlib.pyplot as plt
import numpy as np

def simulate(alpha, beta=0,
             num_periods=8, time_steps_per_period=60):
    # Use oscillations without friction to set dt and T
```

```
    P = 2*np.pi
    dt = P/time_steps_per_period
    T = num_periods*P
    t = np.linspace(0, T, time_steps_per_period*num_periods+1)
    import odespy

    def f(u, t, alpha):
        # Note the sequence of unknowns: v, u (v=du/dt)
        v, u = u
        return [-alpha*np.sign(v) - u, v]

    solver = odespy.RK4(f, f_args=[alpha])
    solver.set_initial_condition([beta, 1])  # sequence must match f
    uv, t = solver.solve(t)
    u = uv[:,1]  # recall sequence in f: v, u
    v = uv[:,0]
    return u, t


if __name__ == '__main__':
    alpha_values = [0, 0.05, 0.1]
    for alpha in alpha_values:
        u, t = simulate(alpha, 0, 6, 60)
        plt.plot(t, u)
        plt.hold('on')
    plt.legend([r'$\alpha=%g$' % alpha for alpha in alpha_values])
    plt.xlabel(r'$\bar t$');  plt.ylabel(r'$\bar u$')
    plt.savefig('tmp.png');   plt.savefig('tmp.pdf')
    plt.show()
```

We find that simulating for 6 periods is relevant for $\alpha = 0, 0.05, 0.1$.



Filename: `sliding_box`.

Problem 2.14: Pendulum equations

The equation for a so-called simple pendulum with a mass m at the end is

$$mL\ddot{\theta} + mg\sin\theta = 0, \tag{2.108}$$

where $\theta(t)$ is the angle with the vertical, L is the length of the pendulum, and g is the acceleration of gravity.

A physical pendulum with moment of inertia I is governed by a similar equation,

$$I\ddot{\theta} + mgL\sin\theta = 0\,. \tag{2.109}$$

Both equations have the initial conditions $\theta(0) = \Theta$ and $\theta'(0) = 0$ (start at rest).

a) Use θ as dimensionless unknown, find a proper time scale, and scale both differential equations.

Solution. Introducing $\bar{t} = t/t_c$ gives

$$mL\frac{1}{t_c^2}\frac{d^2\theta}{d\bar{t}^2} + mg\sin\theta = 0,$$
$$I\frac{1}{t_c^2}\frac{d^2\theta}{d\bar{t}^2} + mgL\sin\theta = 0\,.$$

or on dimensionless form,

$$\frac{d^2\theta}{d\bar{t}^2} + \frac{t_c^2 g}{L}\sin\theta = 0,$$
$$\frac{d^2\theta}{d\bar{t}^2} + \frac{t_c^2 mgL}{I}\sin\theta = 0\,.$$

An obvious choice to make the terms equal are $t_c = \sqrt{L/g}$ in the first equation and $t_c = \sqrt{I/(mgL)}$ in the second. These choices are also compatible with the frequencies if the angle is small: $\ddot{\theta} + g/L\theta = 0$ has solution of the type $\sin(\omega t)$ with $\omega = \sqrt{g/L}$, and then $t_c = 1/\omega$ is a natural scale.

The dimensionless equations become equal in this case:

$$\frac{d^2\theta}{d\bar{t}^2} + \sin\theta = 0\,.$$

b) Some may argue that θ is not dimensionless since it is measured in radians. One may introduce a truly dimensionless angle $\bar{\theta} \in [0,1]$. Set up the scaled ODE problem in this case.

Solution. A $\bar{\theta}\in[0,1]$ is obtained by $\bar{\theta}=\theta/\Theta$. The resulting equation, keeping the time scale as in a), then becomes

$$\frac{d^2\bar{\theta}}{d\bar{t}^2}+\Theta^{-1}\sin(\Theta\bar{\theta})=0,$$

with boundary condition $\bar{\theta}(0)=1$. That is, the only parameter Θ either remains in the ODE or in the initial condition.

There is another line of arguing here too, namely that one should choose the time scale such that the two terms (acceleration and gravity) balances. First we get

$$\frac{d^2\bar{\theta}}{d\bar{t}^2}+\frac{t_c^2 g}{L\Theta}\sin(\Theta\bar{\theta})=0\,.$$

Demanding that the coefficient in the second term is also unity, makes Θ part of the time scale:

$$t_c=\sqrt{\frac{L\Theta}{g}}\,.$$

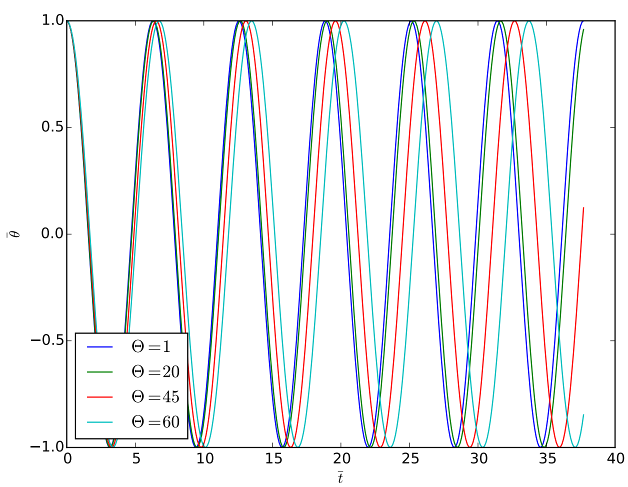
However, we know that in the limit of small θ, doubling the initial condition has no effect on the characteristic time, it only depends on L/g. Therefore this second line of thought will lead to in an appropriate variation of $\bar{\theta}$ with $\bar{t}$. This conclusion can easily be tested through simulations with the two scalings.

c) Simulate the problem in b) for $\Theta=1,20,45,60$ measured in degrees.

Solution. We use Odespy to solve the ODE, rewritten as a system of first-order ODEs: $\bar{\omega}'=-\Theta^{-1}\sin(\Theta\bar{\theta})$ and $\bar{\theta}'=\bar{\omega}$. Appropriate code is

```
import matplotlib.pyplot as plt
import numpy as np

def simulate(Theta, num_periods=8, time_steps_per_period=60,
             scaling=1):
    # Use oscillations for small Theta to set dt and T
    P = 2*np.pi
    dt = P/time_steps_per_period
    T = num_periods*P
    t = np.linspace(0, T, time_steps_per_period*num_periods+1)
    import odespy

    def f(u, t, Theta):
        # Note the sequence of unknowns: omega, theta
        # omega = d(theta)/dt, angular velocity
        omega, theta = u
        return [-Theta**(-1)*np.sin(Theta*theta), omega]

    solver = odespy.RK4(f, f_args=[Theta])
    solver.set_initial_condition([0, 1]) # sequence must match f
    u, t = solver.solve(t)
```

```
    theta = u[:,1]  # recall sequence in f: omega, theta
    return theta, t


if __name__ == '__main__':
    Theta_values_degrees = [1, 20, 45, 60]
    for Theta_degrees in Theta_values_degrees:
        Theta = Theta_degrees*np.pi/180
        theta, t = simulate(Theta, 6, 60)
        plt.plot(t, theta)
        plt.hold('on')
    plt.legend([r'$\Theta=%g$' % Theta
                for Theta in Theta_values_degrees],
               loc='lower left')
    plt.xlabel(r'$\bar t$');  plt.ylabel(r'$\bar\theta$')
    plt.savefig('tmp.png');   plt.savefig('tmp.pdf')
    plt.show()
```



We clearly see that increasing the amplitude Θ increases the period of the oscillations.

Remark. The scaling in b) is more suitable for comparing graphs than the scaling in a) since all the curves have the same amplitude, just different frequency/period. With the scaling in a), we would also get a major difference in amplitudes.
Filename: `pendulum`.

Exercise 2.15: ODEs for a binary star

The equations for a binary star, or a planet and a moon, are

$$m_A \ddot{\boldsymbol{x}}_A = \boldsymbol{F}, \tag{2.110}$$

$$m_B \ddot{\boldsymbol{x}}_B = -\boldsymbol{F}, \tag{2.111}$$

where $\boldsymbol{x}_A$ is the position of object (star) A, and $\boldsymbol{x}_B$ is the position object B. The corresponding masses are m_A and m_B. The only force is the gravity force

$$\boldsymbol{F} = \frac{G m_A m_B}{||\boldsymbol{r}||^3}\boldsymbol{r},$$

where

$$\boldsymbol{r}(t) = \boldsymbol{x}_B(t) - \boldsymbol{x}_A(t),$$

and G is the gravitational constant: $G = 6.674 \cdot 10^{-11}$ Nm^2/kg^2. A problem with these equations is that the parameters are very large (m_A, m_B, $||\boldsymbol{r}||$) or very small (G). The rotation time for binary stars can be very small and large as well.

a) Scale the equations.

Solution. A natural length scale could be the initial distance between the objects: $L = \boldsymbol{r}(0)$. We write the dimensionless quantities as

$$\bar{\boldsymbol{x}}_A = \frac{\boldsymbol{x}_A}{L}, \quad \bar{\boldsymbol{x}}_B = \frac{\boldsymbol{x}_B}{L}, \quad \bar{t} = \frac{t}{t_c}\,.$$

The gravity force is transformed to

$$\boldsymbol{F} = \frac{G m_A m_B}{L^2 ||\bar{\boldsymbol{r}}||^3}\bar{\boldsymbol{r}}, \quad \bar{\boldsymbol{r}} = \bar{\boldsymbol{x}}_B - \bar{\boldsymbol{x}}_A,$$

so the first ODE for $\boldsymbol{x}_A$ becomes

$$\frac{d^2 \bar{\boldsymbol{x}}_A}{d\bar{t}^2} = \frac{G m_B t_c^2}{L^3} \frac{\bar{\boldsymbol{r}}}{||\bar{\boldsymbol{r}}||^3}\,.$$

Assuming that quantities with a bar and their derivatives are around unity in size, it is natural to choose t_c such that the fraction $G m_B t_c / L^2 = 1$:

$$t_c = \sqrt{\frac{L^3}{G m_B}}\,.$$

From the other equation for $\boldsymbol{x}_B$ we get another candidate for t_c with m_A instead of m_B. Which mass we choose play a role if $m_A \ll m_B$ or $m_B \ll m_A$. One solution is to use the sum of the masses:

$$t_c = \sqrt{\frac{L^3}{G(m_A + m_B)}}\,.$$

Taking a look at Kepler's laws of planetary motion, the orbital period for a planet around the star is given by the t_c above, except for a missing factor of 2π, but that means that t_c^{-1} is just the angular frequency of the motion. Our characteristic time t_c is therefore highly relevant. Introducing the dimensionless number

$$\alpha = \frac{m_A}{m_B},$$

we can write the dimensionless ODE as

$$\frac{d^2\bar{\boldsymbol{x}}_A}{d\bar{t}^2} = \frac{1}{1+\alpha}\frac{\bar{\boldsymbol{r}}}{||\bar{\boldsymbol{r}}||^3}, \tag{2.112}$$

$$\frac{d^2\bar{\boldsymbol{x}}_B}{d\bar{t}^2} = \frac{1}{1+\alpha^{-1}}\frac{\bar{\boldsymbol{r}}}{||\bar{\boldsymbol{r}}||^3}\,. \tag{2.113}$$

In the limit $m_A \ll m_B$, i.e., $\alpha \ll 1$, object B stands still, say $\bar{\boldsymbol{x}}_B = 0$, and object A orbits according to

$$\frac{d^2\bar{\boldsymbol{x}}_A}{d\bar{t}^2} = -\frac{\bar{\boldsymbol{x}}_A}{||\bar{\boldsymbol{x}}_A||^3}\,.$$

To better see the motion, and that our scaling is reasonable, we introduce polar coordinates r and θ:

$$\bar{\boldsymbol{x}}_A = r\cos\theta\boldsymbol{i} + r\sin\theta\boldsymbol{j},$$

which means $\bar{\boldsymbol{x}}_A$ can be written as $\bar{\boldsymbol{x}}_A = r\boldsymbol{i}_r$. Since

$$\frac{d}{dt}\boldsymbol{i}_r = \dot{\theta}\boldsymbol{i}_\theta, \quad \frac{d}{dt}\boldsymbol{i}_\theta = -\dot{\theta}\boldsymbol{i}_r,$$

we have

$$\frac{d^2\bar{\boldsymbol{x}}_A}{d\bar{t}^2} = (\ddot{r} - r\dot{\theta}^2)\boldsymbol{i}_r + (r\ddot{\theta} + 2\dot{r}\dot{\theta})\boldsymbol{i}_\theta\,.$$

The equation of motion for mass A is then

$$\ddot{r} - r\dot{\theta}^2 = -\frac{1}{r^2},$$

$$r\ddot{\theta} + 2\dot{r}\dot{\theta} = 0\,.$$

The special case of circular motion, $r = 1$, fulfills the equations, since the latter equation then gives $\dot{\theta} = \text{const}$ and the former then gives $\dot{\theta} = 1$, i.e., the motion is $r(t) = 1$, $\theta(t) = t$, with unit angular frequency as expected and period 2π as expected.

b) Solve the scaled equations numerically for two cases:

1. a planet around a star: $\alpha = 10^{-3}$, $\boldsymbol{x}_A(0) = (1,0)$, $\dot{\boldsymbol{x}}_A(0) = (0,1)$, $\boldsymbol{x}_B(0) = 0$, $\dot{\boldsymbol{x}}_B(0) = 0$
2. two stars: $\alpha = \frac{1}{2}$, $\boldsymbol{x}_A(0) = (1,0)$, $\dot{\boldsymbol{x}}_A(0) = (0,\frac{1}{2})$, $\boldsymbol{x}_B(0) = 0$, $\dot{\boldsymbol{x}}_B(0) = (0,-\frac{1}{2})$

An assumption here is that the orbits are co-planar such that they can be taken to lie in the xy plane.

Solution. Here is an appropriate program (using SciTools for simpler animation code than required by Matplotlib):

```
#import matplotlib.pyplot as plt
import scitools.std as plt
import odespy
import numpy as np

def solver(alpha, ic, T, dt=0.05):
    def f(u, t):
        x_A, vx_A, y_A, vy_A, x_B, vx_B, y_B, vy_B = u
        distance3 = np.sqrt((x_B-x_A)**2 + (y_B-y_A)**2)**3
        system = [
            vx_A,
            1/(1.0 + alpha)*(x_B - x_A)/distance3,
            vy_A,
            1/(1.0 + alpha)*(y_B - y_A)/distance3,
            vx_B,
            -1/(1.0 + alpha**(-1))*(x_B - x_A)/distance3,
            vy_B,
            -1/(1.0 + alpha**(-1))*(y_B - y_A)/distance3,
            ]
        return system

    Nt = int(round(T/dt))
    t_mesh = np.linspace(0, Nt*dt, Nt+1)

    solver = odespy.RK4(f)
    solver.set_initial_condition(ic)
    u, t = solver.solve(t_mesh)
    x_A = u[:,0]
    x_B = u[:,2]
    y_A = u[:,4]
    y_B = u[:,6]
    return x_A, x_B, y_A, y_B, t

def demo_circular():
    # Mass B is at rest at the origin,
    # mass A is at (1, 0) with vel. (0, 1)
    ic = [1, 0, 0, 1, 0, 0, 0, 0]
    x_A, x_B, y_A, y_B, t = solver(
        alpha=0.001, ic=ic, T=2*np.pi, dt=0.01)
```

```
    plt.plot(x_A, x_B, 'r2-', y_A, y_B, 'b2-',
             legend=['A', 'B'],
             daspectmode='equal') # x and y axis have same scaling
    plt.savefig('tmp_circular.png')
    plt.savefig('tmp_circular.pdf')
    plt.show()

def demo_two_stars(animate=True):
    # Initial condition
    ic = [0.6, 0, 0, 1,   # star A: velocity (0,1)
          0, 0, 0, -0.5]  # star B: velocity (0,-0.5)
    # Solve ODEs
    x_A, x_B, y_A, y_B, t = solver(
        alpha=0.5, ic=ic, T=4*np.pi, dt=0.05)
    if animate:
        # Animate motion and draw the objects' paths in time
        for i in range(len(x_A)):
            plt.plot(x_A[:i+1], x_B[:i+1], 'r-',
                     y_A[:i+1], y_B[:i+1], 'b-',
                     [x_A[0], x_A[i]], [x_B[0], x_B[i]], 'r2o',
                     [y_A[0], y_A[i]], [y_B[0], y_B[i]], 'b4o',
                     daspectmode='equal',  # axes aspect
                     legend=['A', 'B', 'A', 'B'],
                     axis=[-1, 1, -1, 1],
                     savefig='tmp_%04d.png' % i,
                     title='t=%.2f' % t[i])
    else:
        # Make a simple static plot of the solution
        plt.plot(x_A, x_B, 'r-', y_A, y_B, 'b-',
                 daspectmode='equal', legend=['A', 'B'],
                 axis=[-1, 1, -1, 1], savefig='tmp_two_stars.png')
    #plt.axes().set_aspect('equal')  # mpl
    plt.show()

if __name__ == '__main__':
    import sys
    if sys.argv[1] == 'circular':
        demo_circular()
    else:
        demo_two_stars(True)
    raw_input()
```

We remark that the sequence of unknowns in `u` must be different if the `odespy.EulerCromer` solver is to be chosen. In that case, the velocity for each degree of freedom must appear before the position.
Filename: `binary_star`.

Problem 2.16: Duffing's equation

Duffing's equation is a vibration equation with linear and cubic spring terms:

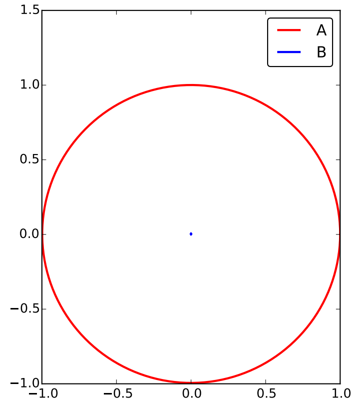


Fig. 2.14 Planet in circular motion around a star.

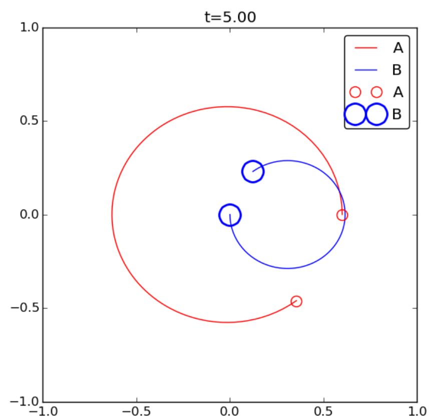


Fig. 2.15 Two rotating stars.

$$mu'' + k_0 u + k_1 u^3 = 0, \quad u(0) = U_0,\ u'(0) = 0\,.$$

Scale this problem.

Solution. We introduce $\bar{t} = t/t_c$ and $\bar{u} = u/u_c$:

$$mt_c^{-2} u_c \bar{u}'' + k_0 u_c \bar{u} + k_1 u_c^3 \bar{u}^3 =, \quad u_c \bar{u}(0) = U_0,\ u_c \bar{u}'(0) = 0\,.$$

Choosing t_c as in a linear vibration problem, $t_c = \sqrt{m/k_0}$, and $u_c = U_0$, we get

$$\bar{u}'' + \bar{u} + \alpha u^3 = 0, \quad \bar{u}(0) = 1,\ \bar{u}'(0) = 0,$$

where

$$\alpha = U_0^2 \frac{k_1}{k_0},$$

is a dimensionless parameter reflecting the ratio of the cubic spring term $k_1 U_0^3$ and the linear spring term $k_0 U_0$ at maximum displacement.
Filename: `Duffing_eq`.

Problem 2.17: Vertical motion in a varying gravity field

A body (e.g., projectile or rocket) is launched vertically from the surface of the earth with velocity V. The body's distance (height) from the earth's surface at time t is represented by the function $h(t)$. Unless h is very much smaller than the earth's radius R, the motion takes place in a varying gravity field. The governing ODE problem for $h(t)$ is then

$$h''(t) = -\frac{R^2 g}{(h+R)^2}, \quad h(0) = 0,\ h'(0) = V, \quad t \in (0, T], \tag{2.114}$$

where g is the acceleration of gravity at the earth's surface.

The goal is to discuss three scalings of this problem. First we introduce

$$\bar{h} = \frac{h}{h_c}, \quad \bar{t} = \frac{t}{t_c},$$

which gives the dimensionless ODE

$$\frac{d^2\bar{h}}{d\bar{t}^2} = -\frac{t_c^2}{h_c}\frac{R^2 g}{(h_c\bar{h} + R)^2} = -\frac{t_c^2}{h_c^3}\frac{R^2 g}{\left(\bar{h} + R/h_c\right)^2}$$

and the dimensionless initial condition

$$\frac{d\bar{h}}{d\bar{t}}(0) = \frac{t_c V}{h_c}.$$

The key dimensionless variable in this problem turns out to be

$$\epsilon = \frac{V}{\sqrt{Rg}}.$$

a) Assume we study the motion over long distances such that h may be of the same size as R. In this case, $h_c = R$ is a reasonable choice. Determine t_c from requiring the initial velocity to be unity. Set up the dimensionless ODE problem.

Solution. The suggested requirement leads to

$$\frac{t_c V}{h_c} = \frac{t_c V}{R} = 1 \quad \Rightarrow \quad t_c = \frac{R}{V}\,.$$

Inserting this t_c in the ODE gives the scaled ODE problem

$$\frac{d^2\bar{h}}{d\bar{t}^2} = -\frac{1}{\epsilon^2\left(1+\bar{h}\right)^2}, \quad \bar{h}(0)=0,\ \frac{d\bar{h}}{d\bar{t}}(0) = 1\,.$$

b) As a), but determine t_c by demanding both terms in the scaled ODE to have unit coefficients.

Solution. We set

$$\frac{t_c^2}{h_c^3}R^2 g = \frac{t_c^2 g}{R} = 1 \quad \Rightarrow \quad t_c = \sqrt{\frac{R}{g}}\,.$$

The initial condition then becomes

$$\frac{d\bar{h}}{d\bar{t}}(0) = \frac{V}{\sqrt{Rg}} = \epsilon\,.$$

The scaled ODE problem is now

$$\frac{d^2\bar{h}}{d\bar{t}^2} = -\frac{1}{\left(1+\bar{h}\right)^2}, \quad \bar{h}(0)=0,\ \frac{d\bar{h}}{d\bar{t}}(0) = \epsilon\,.$$

c) For small initial velocity V, h will be small compared to R. In the limit $h/R \rightarrow 0$, the governing equation simplifies to the well-known motion in a constant gravity field: $h'' = -g$. Use this model to suggest a time and length scale, and derive a dimensionless ODE problem.

Solution. The solution of $h'' = -g$ with $h(0)$ and $h'(0) = V$ is easily obtained by integrating twice: $h = -\frac{1}{2}gt^2 + Vt$. The maximum height reached by the body is found by setting $h'(t) = 0$: $V - gt = 0$, which suggests a corresponding characteristic time $t_c = V/g$. The responding maximum height $h(t_c) = \frac{1}{2}V^2/g$ can be used as characteristic height: $h_c = \frac{1}{2}V^2/g$. The factor $\frac{1}{2}$ is not important, and the ODE problem looks nicer without it, so we simply set $h_c = V^2/g$. Inserted in the initial condition, we get

$$\frac{d\bar{h}}{d\bar{t}}(0) = \frac{Vt_c}{h_c} = 1\,.$$

The scaled ODE takes the form

$$\frac{d^2\bar{h}}{d\bar{t}^2} = -\frac{\epsilon^{-4}}{\left(\bar{h}+2\epsilon^{-2}\right)^2} = -\frac{1}{\left(1+\epsilon^2\bar{h}\right)^2}\,.$$

d) Give an interpretation of the dimensionless parameter ϵ.

Solution. We know from c) that the characteristic height in the constant gravity limit is V^2/g. We can therefore write

$$\epsilon^2 = \frac{V^2/g}{R},$$

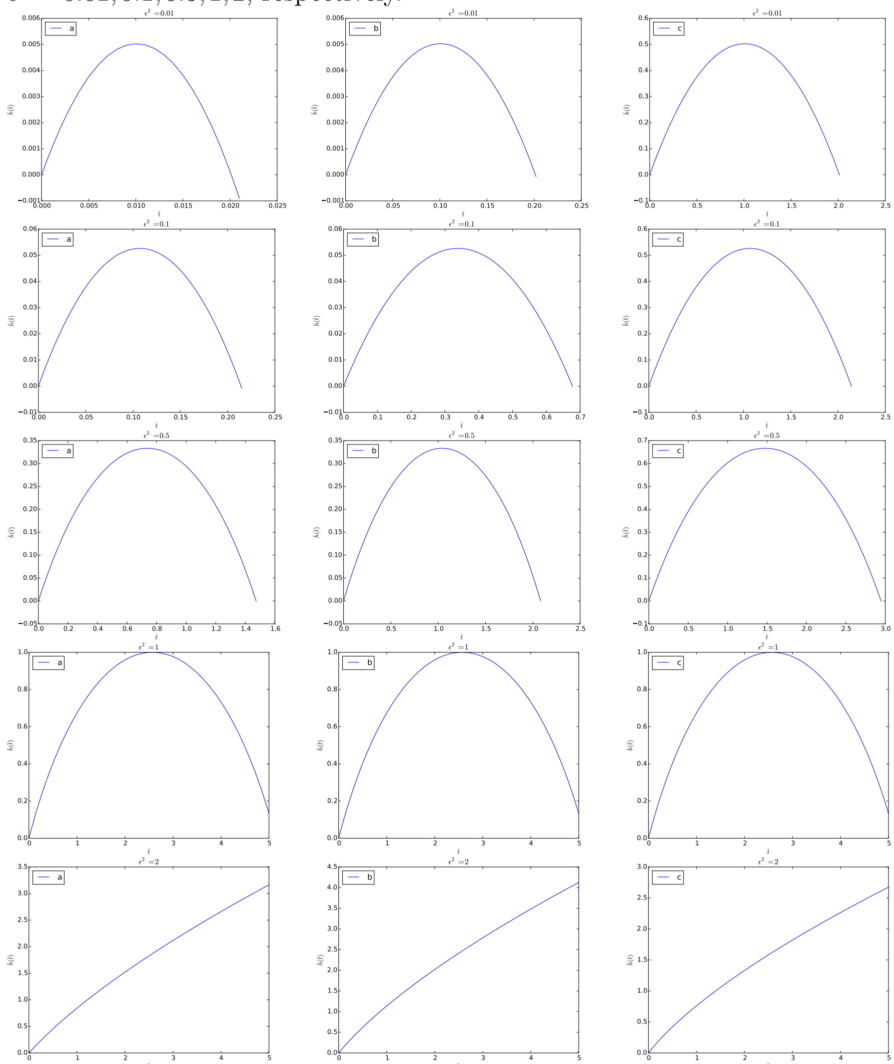
which shows that ϵ^2 is the ratio of the height for small V, i.e., motion in a constant gravity field, and the earth's radius. A small ϵ means that we can neglect varying gravity.

e) Solve numerically for $\bar{h}(\bar{t})$ in each of the three scalings in a), b), and c), with $\epsilon^2 = 0.01, 0.1, 0.5, 1, 2$. When are the various scalings appropriate? (That is, when are $\bar{t}$ and $\bar{h}$ of size unity or at least not very small or big?)

Solution. For numerical solution we rewrite the ODE as a system of two first-order ODEs by introducing a new variable $\bar{v}$ (velocity): $\bar{h}' = \bar{v}$, $\bar{v}' = \ldots$. Here is a code that employs Odespy to solve the system of first-order ODEs:

```
import odespy, numpy as np
import matplotlib.pyplot as plt

def varying_gravity(epsilon):
    def ode_a(u, t):
        h, v = u
        return [v, -epsilon**(-2)/(1 + h)**2]

    def ode_b(u, t):
        h, v = u
        return [v, -1.0/(1 + h)**2]

    def ode_c(u, t):
        h, v = u
        return [v, -1.0/(1 + epsilon**2*h)**2]

    problems = [ode_a, ode_b, ode_c]     # right-hand sides
    ics = [[0, 1], [0, epsilon], [0, 1]] # initial conditions
    for problem, ic, legend in zip(problems, ics, ['a', 'b', 'c']):
        solver = odespy.RK4(problem)
        solver.set_initial_condition(ic)
        t = np.linspace(0, 5, 5001)
        # Solve ODE until h < 0 (h is in u[:,0])
        u, t = solver.solve(t, terminate=lambda u, t, n: u[n,0] < 0)
        h = u[:,0]

        plt.figure()
        plt.plot(t, h)
        plt.legend(legend, loc='upper left')
        plt.title(r'$\epsilon^2=%g$' % epsilon**2)
        plt.xlabel(r'$\bar t$');  plt.ylabel(r'$\bar h(\bar t)$')
        plt.savefig('tmp_%s.png' % legend)
        plt.savefig('tmp_%s.pdf' % legend)
```

Recall from d) that ϵ^2 is the ratio of the height reached in a constant gravity field and the earth's radius. The figures below show the results for $\epsilon^2 = 0.01, 0.1, 0.5, 1, 2$, respectively.



For ϵ small ($\epsilon^2 = 0.01, 0.1$), we see that the scaling in c) is most relevant since the scalings in a) and b) give small $\bar{h}$ and $\bar{t}$. For $\epsilon = 1$, all three scalings are equal. For larger ϵ, the body does not return to the earth. The scalings in a) and c) become equal in the limit $\epsilon \to \infty$, but they are already quite similar for $\epsilon = \sqrt{2}$ according to the bottom figure above. The scaling in c) is therefore the most appealing one since it works for small as well as large ϵ and become close to the others for ϵ around unity.

Remark. The present problem is one of the few problems that is discussed at length in the literature, see Logan [9] or Lin and Segel [8]. The standard argument is that the scaling in c) is favorable since it is the only scaling that is valid as $\epsilon \to 0$. However, that it is robust also for the larger relevant values of ϵ is something that is only clear when we solve the three problems numerically.
Filename: `varying_gravity`.

Problem 2.18: A simplified Schroedinger equation

A simplified stationary Schroedinger's equation for one electron, assuming radial symmetry, takes the form

$$-\frac{\hbar^2}{2m}\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{d}{dr}\right)R + V(r)R = ER, \tag{2.115}$$

where r is the radial coordinate, R is the wave function, $\hbar$ is Planck's constant, m is the mass of the electron, $V =$ is the force potential, which is here taken as the Coulomb potential $V(r) = e^2/(8\pi\epsilon_0 r)$ (where e is the charge of the electron and ϵ_0 is the permittivity of free space), and E is the eigenvalue, for the energy, to be determined along with $R(r)$.

Show that the scaled version of (2.115) can be written

$$-\left(\frac{1}{\bar{r}^2}\frac{d}{d\bar{r}}\bar{r}^2\frac{d}{d\bar{r}}\right)\bar{R} + \frac{1}{\bar{r}}\bar{R} = \lambda\bar{R}, \tag{2.116}$$

where λ is a dimensionless eigenvalue

$$\lambda = \frac{(4\pi)^2\epsilon_0^2\hbar^2 E}{me^4}\,.$$

The symbol $\bar{r}$ is the scaled coordinate, and $\bar{R}$ is a scaled version of R (the scaling factor drops out of the equation). The length scale, which arises naturally, is the Bohr radius.

Solution. We introduce

$$\bar{r} = \frac{r}{r_c}, \quad \bar{R} = \frac{R}{R_c},$$

and insert these expressions in the differential equation. Multiplying with $2mr_c^2/\hbar^2$ we get

$$-\left(\frac{1}{\bar{r}^2}\frac{d}{d\bar{r}}\left(\bar{r}^2\frac{d}{d\bar{r}}\right)\right)\bar{R} + \frac{me^2 r_c}{4\pi\epsilon_0\hbar^2}\frac{1}{\bar{r}}\bar{R} = \frac{2Emr_c^2}{\hbar^2}R\,.$$

Note that all the R_c factors cancel.

Balance of the two terms on the left-hand side suggests that the length scale r_c can be determined from requiring

$$\frac{me^2 r_c}{4\pi\epsilon_0\hbar^2} = 1,$$

i.e.,

$$r_c = \frac{4\pi\epsilon_0\hbar^2}{me^2},$$

which is actually the Bohr radius, demonstrating that the balance of the terms on the left-hand side automatically determines a very relevant length scale.

A bit of arithmetics in the right-hand side term gives the given expression for λ. We then end up with (2.116).
Filename: `Schroedinger`.

Remarks. Introducing $u = \bar{r}\bar{R}$ and renaming $\bar{r}$ to x, (2.116) can be recast in the simpler form

$$-u''(x) + \frac{1}{x}u(x) = \lambda u(x),$$

which is a simpler eigenvalue problem to solve numerically (the boundary conditions are $u(0) = 0$ and $\lim_{x\to\infty} u(x) = 0$).

Chapter 3
Basic partial differential equation models

This chapter extends the scaling technique to well-known partial differential equation (PDE) models for waves, diffusion, and transport. We start out with the simplest 1D models of the PDEs and then progress with additional terms, different types of boundary and initial conditions, and generalizations to 2D and 3D.

3.1 The wave equation

A standard, linear, one-dimensional wave equation problem in a homogeneous medium may be written as

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}, \quad x \in (0, L),\ t \in (0, T], \tag{3.1}$$

where c is the constant wave velocity of the medium. With a briefer notation, where subscripts indicate derivatives, the PDE (3.1) can be written $u_{tt} = c^2 u_{xx}$. This subscript notation will occasionally be used later.

For any number of dimensions in heterogeneous media we have the generalization

$$\frac{\partial^2 u}{\partial t^2} = \nabla \cdot \left(c^2 \nabla u\right) + f, \quad x, y, z \in \Omega,\ t \in (0, T], \tag{3.2}$$

where f represents a forcing.

3.1.1 Homogeneous Dirichlet conditions in 1D

Let us first start with (3.1), homogeneous Dirichlet conditions in space, and no initial velocity u_t:

$$u(x,0) = I(x), \qquad x \in [0,L], \tag{3.3}$$
$$\frac{\partial}{\partial t}u(x,0) = 0, \qquad x \in [0,L], \tag{3.4}$$
$$u(0,t) = 0, \qquad t \in (0,T], \tag{3.5}$$
$$u(L,t) = 0, \qquad t \in (0,T]. \tag{3.6}$$

The independent variables are x and t, while u is the dependent variable. The rest of the parameters, c, L, T, and $I(x)$, are given data.

We start with introducing dimensionless versions of the independent and dependent variables:

$$\bar{x} = \frac{x}{x_c}, \quad \bar{t} = \frac{t}{t_c}, \quad \bar{u} = \frac{u}{u_c}\,.$$

Inserting the $x = x_c\bar{x}$, etc., in (3.1) and (3.3)-(3.6) gives

$$\begin{aligned}
\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} &= \frac{t_c^2 c^2}{x_c^2}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2}, && \bar{x} \in (0, L/x_c),\ \bar{t} \in (0, T/t_c],\\
\bar{u}(\bar{x},0) &= \frac{I(x_c\bar{x})}{u_c}, && \bar{x} \in [0, L/x_c],\\
\frac{\partial}{\partial \bar{t}}\bar{u}(\bar{x},0) &= 0, && \bar{x} \in [0, L/x_c],\\
\bar{u}(0,\bar{t}) &= 0, && \bar{t} \in (0, T/t_c],\\
\bar{u}(L/x_c,\bar{t}) &= 0, && \bar{t} \in (0, T/t_c].
\end{aligned}$$

The key question is how to define the scales. A natural choice is $x_c = L$ since this makes $\bar{x} \in [0,1]$. For the spatial scale and the problem governed by (3.1) we have some analytical insight that can help. The solution behaves like

$$u(x,t) = f_R(x - ct) + f_R(x + ct), \tag{3.7}$$

i.e., a right- and left-going wave with velocity c. The initial conditions constrain the choices of f_R and f_L to $f_L + f_R = I$ and $-cf_L' + cf_R' = 0$. The solution is $f_R = f_L = \frac{1}{2}$, and consequently

$$u(x,t) = \frac{1}{2}I(x - ct) + \frac{1}{2}I(x + ct),$$

which tells that the initial condition splits in two, half of it moves to the left and half to the right. This means in particular that we can choose $u_c = \max_x |I(x)|$ and get $|\bar{u}| \le 1$, which is a goal. It must be added that boundary conditions may result in reflected waves, and the solution is then more complicated than indicated in the formula above.

Regarding the time scale, we may look at the two terms in the scaled PDE and argue that if $|u|$ and its derivatives are to be of order unity, then the size of the second-order derivatives should be the same, and t_c can be chosen to make the coefficient $t_c^2 c^2/x_c^2$ unity, i.e., $t_c = L/c$. Another reasoning may set t_c as the time it takes the wave to travel through the domain $[0,L]$. Since the wave has constant speed c, $t_c = L/c$.

With the described choices of scales, we end up with the dimensionless initial-boundary value problem

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} = \frac{\partial^2 \bar{u}}{\partial x^2}, \qquad \bar{x} \in (0,1),\ \bar{t} \in (0,\bar{T}], \tag{3.8}$$

$$\bar{u}(\bar{x},0) = \frac{I(\bar{x}L)}{\max_{x\in(0,L)}|I(x)|}, \qquad \bar{x} \in [0,1], \tag{3.9}$$

$$\frac{\partial}{\partial \bar{t}}\bar{u}(\bar{x},0) = 0, \qquad \bar{x} \in [0,1], \tag{3.10}$$

$$\bar{u}(0,\bar{t}) = 0, \qquad \bar{t} \in (0,\bar{T}], \tag{3.11}$$

$$\bar{u}(1,\bar{t}) = 0, \qquad \bar{t} \in (0,\bar{T}]. \tag{3.12}$$

Here, $\bar{T} = Tc/L$.

The striking feature of (3.8)-(3.12) is that there are *no physical parameters* involved! Everything we need to specify is the shape of the initial condition and then scale it such that it is less than or equal to 1.

The physical solution with dimension is recovered from $\bar{u}(\bar{x},\bar{t})$ through

$$u(x,t) = \max_{x\in(0,L)} I(x)\,\bar{u}(\bar{x}L,\bar{t}L/c) \tag{3.13}$$

3.1.2 Implementation of the scaled wave equation

How do we implement (3.8)-(3.12)? As for the simpler mathematical models, we suggest to implement the model with dimensions and observe how to set parameters to obtain the scaled model. In the present case, one must choose $L = 1$, $c = 1$, and scale I by its maximum value. That's all!

Several implementations of 1D wave equation models with different degree of mathematical and software complexity come along with these notes. The simplest version is `wave1D_u0.py` that implements (3.1) and (3.3)-(3.6). This is the code to be used in the following. It is described in Section 2.3 in [7].

Waves on a string. As an example, we may let the original initial-boundary value problem (3.1)-(3.6) model vibrations of a string on a string instrument (e.g., a guitar). With u as the displacement of the string, the boundary conditions $u = 0$ at the ends are relevant, as well as the zero velocity condition $\partial u/\partial t = 0$ at $t = 0$. The initial condition $I(x)$ typically has a triangular shape for a picked guitar string. The physical problem needs parameters for the amplitude of $I(x)$, the length L of the string, and the value of c for the string. Only the latter is challenging as it involves relating c to the pitch (i.e., time frequency) of the string. In the scaled problem, we can forget about all this. We simply set $L=1$, $c=1$, and let $I(x)$ have a peak of unity at $x=x_0\in(0,1)$:

$$\frac{I(x)}{\max_x I(x)} = \begin{cases} x/x_0, & x < x_0,\\ (1-x)/(1-x_0), & \text{otherwise}\end{cases}$$

The dimensionless coordinate of the peak, x_0, is the only dimensionless parameter in the problem. For fixed x_0, one single simulation will capture all possible solutions with such an initial triangular shape.

Detecting an already computed case. The file `wave1D_u0_scaled.py` has functionality for detecting whether a simulation corresponds to a previously run scaled case, and if so, the solution is retrieved from file. The implementation technique makes use of `joblib`, but is more complicated than shown previously in these notes since some of the arguments to the function that computes the solution are functions, and one must recognized if the function has been used as argument before or not. There is documentation in the `wave1D_u0_scaled.py` file explaining how this is done.

3.1.3 Time-dependent Dirichlet condition

A generalization of (3.1)-(3.6) is to allow for a time-dependent Dirichlet condition at one end, say $u(0,t)=U_L(t)$. At the other end we may still have $u=0$. This new condition at $x=0$ may model a specified wave that enters the domain. For example, if we feed in a monochromatic wave $A\sin(k(x-ct))$ from the left end, $U_L(t)=A\sin(kct)$. This forcing of the wave motion has its own amplitude and time scale that could affect the choice of u_c and t_c.

The main difference from the previous initial-boundary value problem is the condition at $x=0$, which now reads

$$\bar{u}(0,\bar{t}) = \frac{U_L(\bar{t}t_c)}{u_c}$$

in scaled form.

Scaling. Regarding the characteristic time scale, it is natural to base this scale on the wave propagation velocity, together with the length scale, and not on the time scale of $U_L(t)$, because the time scale of U_L basically determines whether short or long waves are fed in at the boundary. All waves, long or short, propagate with the same velocity c. We therefore continue to use $t_c=L/c$.

The solution u will have one wave contribution from the initial condition I and one from the feeding of waves at $x=0$. This gives us three choices of u_c:

$\max_x|I|+\max_t|U_L|$, $\max_x|I|$, or $\max_t|U_L|$. The first seems relevant if the size of I and U_L are about the same, but then we can choose either $\max_x|I|$ or $\max_t|U_L|$ as characteristic size of u since a factor of 2 is not important. If I is much less than U_L, $u_c=\max_t|u_L|$ is relevant, while $u_c=\max_x|I|$ is the choice when I has much bigger impact than U_L on u.

With $u_c=\max_t|U_L(t)|$, we get the scaled problem

$$\frac{\partial^2\bar{u}}{\partial\bar{t}^2}=\frac{\partial^2\bar{u}}{\partial\bar{x}^2},\qquad \bar{x}\in(0,1),\ \bar{t}\in(0,\bar{T}], \tag{3.14}$$

$$\bar{u}(\bar{x},0)=\frac{I(x_c\bar{x})}{\max_t|U_L(t)|},\qquad \bar{x}\in[0,1], \tag{3.15}$$

$$\frac{\partial}{\partial\bar{t}}\bar{u}(\bar{x},0)=0,\qquad \bar{x}\in[0,1], \tag{3.16}$$

$$\bar{u}(0,\bar{t})=\frac{U_L(\bar{t}t_c)}{\max_t|U_L(t)|},\qquad \bar{t}\in(0,\bar{T}], \tag{3.17}$$

$$\bar{u}(1,\bar{t})=0,\qquad \bar{t}\in(0,\bar{T}]. \tag{3.18}$$

Also this problem is free of physical parameters like c and L. The input is completely specified by the shape of $I(x)$ and $U_L(t)$.

Software. Software for the original problem with dimensions can be reused for (3.14)-(3.18) by setting $L=1$, $c=1$, and scaling $U_L(t)$ and $I(x)$ by $\max_t|U_L(t)|$.

Specific case. As an example, consider

$$\begin{aligned}U_L(t)&=a\sin(\omega t)\text{ for }0\le t\le 2\frac{\omega}{2\pi},\text{ else }0,\\ I(x)&=Ae^{-(x-L/2)^2/\sigma^2}.\end{aligned}$$

That is, we start with a Gaussian peak-shaped wave in the center of the domain and feed in a sinusoidal wave at the left end for two periods. The solution will be the sum of three waves: two parts from the initial condition, plus the wave fed in from the left.

Since $\max_t|U_L|=a$ we get

$$\bar{u}(\bar{x},0)=\frac{A}{a}e^{-(L/\sigma)^2(\bar{x}-\frac{1}{2})^2}, \tag{3.19}$$

$$\bar{u}(0,\bar{t})=\sin(\bar{t}\omega L/c). \tag{3.20}$$

Here, U_L models an incoming wave $a\sin(k(x-ct)$, with k specified. The result is incoming waves of length $\lambda=2\pi/k$. Since $\omega=kc$, $\bar{u}(0,\bar{t})=\sin(kL\bar{t})=\sin(2\pi\bar{t}L/\lambda)$. (This formula demonstrates the previous assertion that the time scale of U_L, i.e., $1/\omega$, determines the wave length $1/\omega=\lambda/(2\pi)$ in space.) We realize from the formulas (3.19) and (3.20) that there are three key dimensionless parameters related to these specific choices of initial and boundary conditions:

$$\alpha = \frac{A}{a}, \quad \beta = \frac{L}{\sigma}, \quad \gamma = kL = 2\pi\frac{L}{\lambda}\,.$$

With α, β, and γ we can write the dimensionless initial and boundary conditions as

$$\begin{aligned}
\bar u(\bar x, 0) &= \alpha e^{-\beta^2(\bar x - \frac{1}{2})^2},\\
\bar u(0, \bar t) &= \sin(\gamma\bar t)\,.
\end{aligned}$$

The dimensionless parameters have the following interpretations:

- α: ratio of initial condition amplitude and amplitude of incoming wave at $x=0$
- β: ratio of length of domain and width of initial condition
- γ: ratio of length of domain and wave length of incoming wave

Again, these dimensionless parameters tell a lot about the interplay of the physical effects in the problem. And only some ratios count!

We can simulate two special cases:

1. $\alpha = 10$ (large) where the incoming wave is small and the solution is dominated by the two waves arising from $I(x)$,
2. $\alpha = 0.1$ (small) where the incoming waves dominate and the solution has the initial condition just as a small perturbation of the wave shape.

We may choose a peak-shaped initial condition: $\beta = 10$, and also a relatively short incoming wave compared to the domain size: $\gamma = 6\pi$ (i.e., wave length of incoming wave is $L/6$). A function `simulate_Gaussian_and_incoming_wave` in the file `session.py` applies the general unscaled solver in `wave1D_dn.py` for solving the wave equation with constant c, and any time-dependent function or $\partial u/\partial x = 0$ at the end points. This solver is trivially adapted to the present case. Figures 3.1 and 3.2 shows snapshots of how $\bar u(\bar x, \bar t)$ evolves due to a large/small initial condition and small/large incoming wave at the left boundary.

Movie 1: $\alpha = 10$. `https://github.com/hplgit/scaling-book/raw/master/doc/pub/book/html/mov-scaling/gaussian_plus_incoming/alpha10.mp4`

Movie 2: $\alpha = 0.1$. `https://github.com/hplgit/scaling-book/raw/master/doc/pub/book/html/mov-scaling/gaussian_plus_incoming/alpha01.mp4`

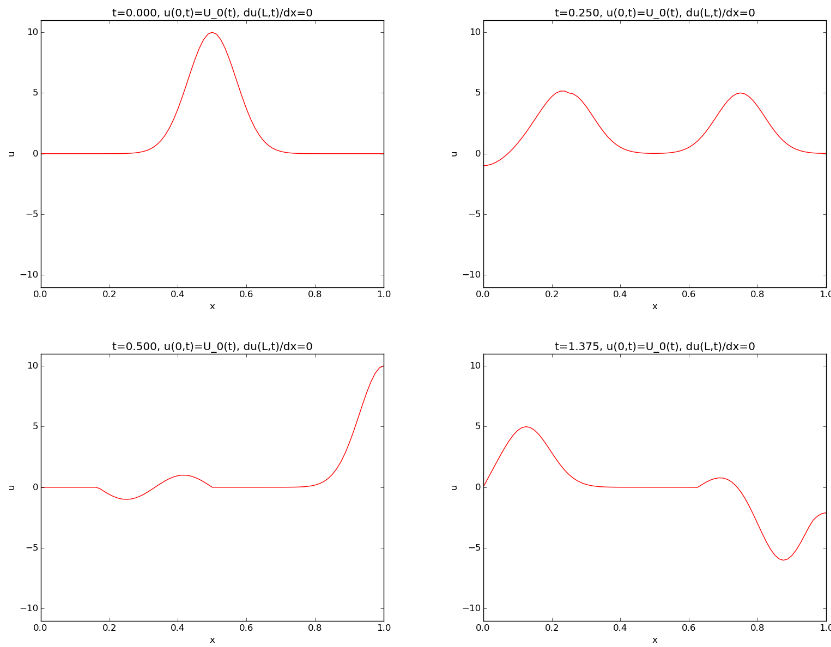


Fig. 3.1 Snapshots of solution with large initial condition and small incoming wave ($\alpha = 10$).

3.1.4 Velocity initial condition

Now we change the initial condition from $u = I$ and $\partial u/\partial t = 0$ to

$$u(x,0) = 0, \tag{3.21}$$

$$\frac{\partial}{\partial t}u(x,0) = V(x)\,. \tag{3.22}$$

Impact problems are often of this kind. The scaled version of $u_t(x,0) = V(x)$ becomes

$$\frac{\partial}{\partial \bar{t}}\bar{u}(\bar{x},0) = \frac{t_c}{u_c}V(\bar{x}x_c)\,.$$

Analytical insight. From (3.7) we now get $f_L + f_R = 0$ and $cf_L' - cf_R' = V$. Introducing $W(x)$ such that $W'(x) = V(x)$, a solution is $f_L = \frac{1}{2}W/c$ and $-f_R = \frac{1}{2}W/c$. We can express this solution through the formula

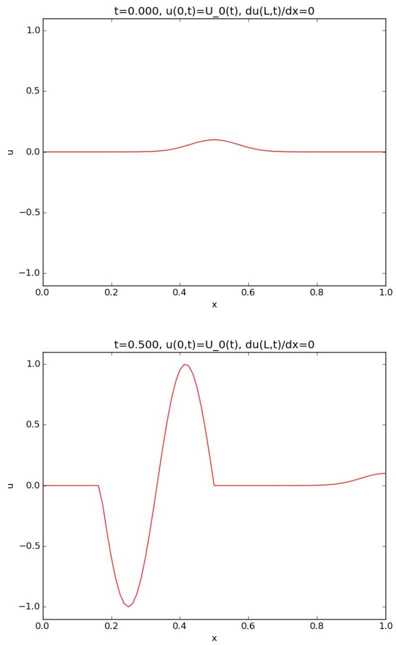


Fig. 3.2 Snapshots of solution with small initial condition and large incoming wave ($\alpha = 0.1$).

$$u(x,t) = \frac{1}{2c}\int_{x-ct}^{x+ct} V(\xi)d\xi = \frac{1}{2c}(W(x+ct) - W(x-ct))\,. \tag{3.23}$$

Scaling. Since V is the time-derivative of u, the characteristic size of V, call it V_c, is typically u_c/t_c. If we, as usual, base t_c on the wave speed, $t_c = L/c$, we get $u_c = V_c L/c$. Looking at the solution (3.23), we see that u_c has size $\text{mean}(V)L/(2c)$, where $\text{mean}(V)$ is the mean value of V ($W \sim \text{mean}(V)L$). This result suggests $V_c = \text{mean}(V)$ and $u_c = \text{mean}(V)L/(2c)$. One may argue that the factor 2 is not important, but if we want $|\bar{u}| \in [0,1]$ it is convenient to keep it.

The scaled initial condition becomes

$$\frac{\partial}{\partial \bar{t}}\bar{u}(\bar{x},0) = \frac{t_c}{u_c}V(\bar{x}x_c) = \frac{V(\bar{x}x_c)}{\frac{1}{2}\text{mean}(V)}\,.$$

Nonzero initial shape. Suppose we change the initial condition $u(x,0) = 0$ to $u(x,0) = I(x)$. The scaled version of this condition with the above u_c based on V becomes

$$\bar{u}(\bar{x},0) = \frac{2cI(\bar{x}x_c)}{L\,\text{mean}(V)}\,. \tag{3.24}$$

Check that dimensionless numbers are dimensionless!

Is a dimensionless number really dimensionless? It is easy to make errors when scaling equations, so checking that such fractions are dimensionless is wise. The dimension of I is the same as u, here taken to be displacement: [L]. Since V is $\partial u/\partial t$, its dimension is $[\mathrm{LT}^{-1}]$. The dimensions of c and L are $[\mathrm{LT}^{-1}]$ and [L]. The dimension of the right-hand side of (3.24) is then

$$\frac{[\mathrm{LT}^{-1}][L]}{[L][L\mathrm{T}^{-1}]} = 1,$$

demonstrating that the fraction is indeed dimensionless.

One may introduce a dimensionless initial shape, $\bar{I}(\bar{x}) = I(\bar{x}L)/\max_x |I|$. Then

$$\bar{u}(\bar{x},0) = \alpha\bar{I}(\bar{x}),$$

where α the dimensionless number

$$\alpha = \frac{2c}{L}\frac{\max_x |I(x)|}{\mathrm{mean}(V)}\,.$$

If V is much larger than I, one expects that the influence of I is small. However, it takes time for the initial velocity V to influence the wave motion, so the speed of the waves c and the length of the domain L also play a role. This is reflected in α, which is the important parameter. Again, the scaling and the resulting dimensionless parameter(s) teach us much about the interaction of the various physical effects.

3.1.5 Variable wave velocity and forcing

The next generalization regards wave propagation in a non-homogeneous medium where the wave velocity c depends on the spatial position: $c = c(x)$. To simplify the notation we introduce $\lambda(x) = c^2(x)$. We introduce homogeneous Neumann conditions at $x = 0$ and $x = L$. In addition, we add a force term $f(x,t)$ to the PDE, modeling wave generation in the interior of the domain. For example, a moving slide at the bottom of a fjord will generate surface waves and is modeled by such an $f(x,t)$ term (provided the length of the waves is much larger than the depth so that a simple wave equation like (3.25) applies). The initial-boundary value problem can be then expressed as

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial}{\partial x}\left(\lambda(x)\frac{\partial u}{\partial x}\right) + f(x,t), \qquad x \in (0,L),\ t \in (0,T], \tag{3.25}$$
$$u(x,0) = I(x), \qquad x \in [0,L], \tag{3.26}$$
$$\frac{\partial}{\partial t}u(x,0) = 0, \qquad x \in [0,L], \tag{3.27}$$
$$\frac{\partial}{\partial x}u(0,t) = 0, \qquad t \in (0,T], \tag{3.28}$$
$$\frac{\partial}{\partial x}u(L,t) = 0, \qquad t \in (0,T]. \tag{3.29}$$

Non-dimensionalization. We make the coefficient λ non-dimensional by

$$\bar{\lambda}(\bar{x}) = \frac{\lambda(\bar{x}x_c)}{\lambda_c}, \tag{3.30}$$

where one normally chooses the characteristic size of λ, λ_c, to be the maximum value such that $|\lambda| \le 1$:

$$\lambda_c = \max_{x\in(0,L)} \lambda(x)\,.$$

Similarly, f has a scaled version

$$\bar{f}(\bar{x},\bar{t}) = \frac{f(\bar{x}x_c,\bar{t}t_c)}{f_c},$$

where normally we choose

$$f_c = \max_{x,t}|f(x,t)|\,.$$

Inserting dependent and independent variables expressed by their non-dimensional counterparts yields

$$\begin{aligned}
\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} &= \frac{t_c^2\lambda_c}{L^2}\frac{\partial}{\partial \bar{x}}\left(\bar{\lambda}(\bar{x})\frac{\partial \bar{u}}{\partial \bar{x}}\right) + \frac{t_c^2 f_c}{u_c}\bar{f}(\bar{x},\bar{t}), && \bar{x} \in (0,1),\ \bar{t} \in (0,\bar{T}],\\
\bar{u}(\bar{x},0) &= \frac{I(x)}{u_c}, && \bar{x} \in [0,1],\\
\frac{\partial}{\partial \bar{t}}\bar{u}(\bar{x},0) &= 0, && \bar{x} \in [0,1],\\
\frac{\partial}{\partial \bar{x}}\bar{u}(0,\bar{t}) &= 0, && \bar{t} \in (0,\bar{T}],\\
\frac{\partial}{\partial \bar{x}}\bar{u}(1,\bar{t}) &= 0, && \bar{t} \in (0,\bar{T}],
\end{aligned}$$

with $\bar{T} = Tc/L$.

Choosing the time scale. The time scale is, as before, chosen as $t_c = L/\sqrt{\lambda_c}$. Note that the previous (constant) wave velocity c now corresponds to $\sqrt{\lambda(x)}$. Therefore, $\sqrt{\lambda_c}$ is a characteristic wave velocity.

One could wonder if the time scale of the force term, $f(x,t)$, should influence t_c, but as we reasoned for the boundary condition $u(0,t) = U_L(t)$, we let the characteristic time be governed by the signal speed in the medium, i.e., by $\sqrt{\lambda_c}$ here and not by the time scale of the excitation f, which dictates the length of the generated waves and not their propagation speed.

Choosing the spatial scale. We may choose u_c as $\max_x |I(x)|$, as before, or we may fit u_c such that the coefficient in the source term is unity, i.e., all terms balance each other. This latter idea leads to

$$u_c = \frac{L^2 f_c}{\lambda_c}$$

and a PDE without parameters,

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} = \frac{\partial}{\partial \bar{x}}\left(\bar{\lambda}(\bar{x})\frac{\partial \bar{u}}{\partial \bar{x}}\right) + \bar{f}(\bar{x},\bar{t})\,.$$

The initial condition $u(x,0) = I(x)$ becomes in dimensionless form

$$\bar{u}(\bar{x},0) = u_c^{-1}\max_x |I(x)|\bar{I}(\bar{x}) = \beta^{-1}\bar{I}(\bar{x}),$$

where

$$\beta = \frac{L^2}{\lambda_c}\frac{\max_{x,t}|f(x,t)|}{\max_x |I(x)|}\,.$$

In the case $u_c = \max_x |I(x)|$, $\bar{u}(\bar{x},0) = \bar{I}(\bar{x})$ and the β parameter appears in the PDE instead:

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} = \frac{\partial}{\partial \bar{x}}\left(\bar{\lambda}(\bar{x})\frac{\partial \bar{u}}{\partial \bar{x}}\right) + \beta\bar{f}(\bar{x},\bar{t})\,.$$

With $V = 0$, and $u = 0$ or $u_x = 0$ on the boundaries $x = 0, L$, this scaling normally gives $|\bar{u}| \leq 1$, since initially $|I| \leq 1$, and no boundary condition can increase the amplitude. However, the forcing, $\bar{f}$, may inherit spatial and temporal scales of its own that may complicate the matter. The forcing may, for instance, be some disturbance moving with a velocity close to the propagation velocity of the free waves. This will have an effect akin to the resonance for the vibration problem discussed in section 2.2.2 and the waves produced by the forcing may be much larger than indicated by β. On the other hand, the forcing may also consist of alternating positive and negative parts (retrogressive slides constitute an example). These may interfere to reduce the wave generation by an order of magnitude.

Scaling the velocity initial condition. The initial condition $u_t(x,0) = V(x)$ has its dimensionless variant as

$$\bar{V}(\bar{x}) = \frac{t_c}{u_c}\frac{V(L\bar{x})}{\max_x |V(x)|},$$

which becomes

$$\frac{\partial \bar{u}}{\partial \bar{t}}(\bar{x},0) = \frac{L}{\sqrt{\lambda_c}}\frac{\max_x |V(x)|}{\max_x |I(x)|}\bar{V}(\bar{x}),\ \text{if } u_c = \max_x |I(x)|,$$

or

$$\frac{\partial \bar{u}}{\partial \bar{t}}(\bar{x},0) = \frac{\sqrt{\lambda_c}}{L}\frac{\max_x |V(x)|}{\max_{x,t} |f(x,t)|}\bar{V}(\bar{x}),\ \text{if } u_c = t_c^2 f_c = \frac{L^2}{\lambda_c}\max_{x,t} |f(x,t)|\,.$$

Introducing the dimensionless number α (cf. Section 3.1.4),

$$\alpha^{-1} = \frac{\sqrt{\lambda_c}}{L}\frac{\max_x |V(x)|}{\max_{x,t} |f(x,t)|},$$

we can write

$$\frac{\partial \bar{u}}{\partial \bar{t}}(\bar{x},0) = \begin{cases} \alpha^{-1}\bar{V}(\bar{x}), & u_c = \max_x |I| \\ \alpha^{-1}\beta^{-1}\bar{V}(\bar{x}), & u_c = t_c^2 f_c \end{cases}$$

3.1.6 Damped wave equation

A linear damping term $b\partial u/\partial t$ is often added to the wave equation to model energy dissipation and amplitude reduction. Our PDE then reads

$$\frac{\partial^2 u}{\partial t^2} + b\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(\lambda(x)\frac{\partial u}{\partial x}\right) + f(x,t)\,. \tag{3.31}$$

The scaled equation becomes

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} + \frac{t_c}{b}\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{t_c^2\lambda_c}{L^2}\frac{\partial}{\partial \bar{x}}\left(\bar{\lambda}(\bar{x})\frac{\partial \bar{u}}{\partial \bar{x}}\right) + \frac{t_c^2 f_c}{u_c}\bar{f}(\bar{x},\bar{t})\,.$$

The damping term is usually much smaller than the two other terms involving $\bar{u}$. The time scale is therefore chosen as in the undamped case, $t_c = L/\sqrt{\lambda_c}$. As in Section 3.1.5, we have two choices of u_c: $u_c = \max_x |I|$ or $u_c = t_c^2 f_c$. The former choice of u_c gives a PDE with two dimensionless numbers,

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} + \gamma\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial}{\partial \bar{x}}\left(\bar{\lambda}(\bar{x})\frac{\partial \bar{u}}{\partial \bar{x}}\right) + \beta\bar{f}(\bar{x},\bar{t}), \tag{3.32}$$

where

$$\gamma = \frac{bL}{\sqrt{\lambda_c}},$$

measures the size of the damping, and β is as given in Section 3.1.5. With $u_c = t_c^2 f_c$ we get a PDE where only γ enters,

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} + \gamma \frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial}{\partial \bar{x}}\left(\bar{\lambda}(\bar{x})\frac{\partial \bar{u}}{\partial \bar{x}}\right) + \bar{f}(\bar{x},\bar{t})\,. \tag{3.33}$$

The scaled initial conditions are as in Section 3.1.5, so in this latter case β appears in the initial condition for u.

To summarize, the effects of V, f, and damping are reflected in the dimensionless numbers α, β, and γ, respectively.

3.1.7 A three-dimensional wave equation problem

To demonstrate how the scaling extends to in three spatial dimensions, we consider

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} = \frac{\partial}{\partial x}\left(\lambda \frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(\lambda \frac{\partial u}{\partial y}\right) + \frac{\partial}{\partial z}\left(\lambda \frac{\partial u}{\partial z}\right). \tag{3.34}$$

Introducing

$$\bar{x} = \frac{x}{x_c}, \quad \bar{y} = \frac{y}{y_c}, \quad \bar{z} = \frac{z}{z_c}, \quad \bar{t} = \frac{t}{t_c}, \quad \bar{u} = \frac{u}{u_c},$$

and scaling λ as $\bar{\lambda} = \lambda(\bar{x}x_c, \bar{y}y_c, \bar{z}z_c)/\lambda_c$, we get

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} = \frac{t_c^2\lambda_c}{x_c^2}\frac{\partial}{\partial \bar{x}}\left(\bar{\lambda} \frac{\partial \bar{u}}{\partial \bar{x}}\right) + \frac{t_c^2\lambda_c}{y_c^2}\frac{\partial}{\partial \bar{y}}\left(\bar{\lambda} \frac{\partial \bar{u}}{\partial \bar{y}}\right) + \frac{t_c^2\lambda_c}{z_c^2}\frac{\partial}{\partial \bar{z}}\left(\bar{\lambda} \frac{\partial \bar{u}}{\partial \bar{z}}\right).$$

Often, we will set $x_c = y_c = z_c = L$ where L is some characteristic size of the domain. As before, $t_c = L/\sqrt{\lambda_c}$, and these choices lead to a dimensionless wave equation without physical parameters:

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} = \frac{\partial}{\partial \bar{x}}\left(\bar{\lambda} \frac{\partial \bar{u}}{\partial \bar{x}}\right) + \frac{\partial}{\partial \bar{y}}\left(\bar{\lambda} \frac{\partial \bar{u}}{\partial \bar{y}}\right) + \frac{\partial}{\partial \bar{z}}\left(\bar{\lambda} \frac{\partial \bar{u}}{\partial \bar{z}}\right). \tag{3.35}$$

The initial conditions remain the same as in the previous one-dimensional examples.

3.2 The diffusion equation

The diffusion equation in a one-dimensional homogeneous medium reads

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \quad x \in (0, L),\ t \in (0, T], \tag{3.36}$$

where α is the diffusion coefficient. The multi-dimensional generalization to a heterogeneous medium and a source term takes the form

$$\frac{\partial u}{\partial t} = \nabla \cdot (\alpha \nabla u) + f, \quad x, y, z \in \Omega,\ t \in (0, T]\,. \tag{3.37}$$

We first look at scaling of the PDE itself, and thereafter we discuss some types of boundary conditions and how to scale the complete initial-boundary value problem.

3.2.1 Homogeneous 1D diffusion equation

Choosing the time scale. To make (3.36) dimensionless, we introduce, as usual, dimensionless dependent and independent variables:

$$\bar{x} = \frac{x}{x_c}, \quad \bar{t} = \frac{t}{t_c}, \quad \bar{u} = \frac{u}{u_c}\,.$$

Inserting the dimensionless quantities in the one-dimensional PDE (3.36) results in

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{t_c \alpha}{L^2} \frac{\partial^2 \bar{u}}{\partial \bar{x}^2}, \quad \bar{x} \in (0, 1),\ \bar{t} \in (0, \bar{T} = T/t_c]\,.$$

Arguing, as for the wave equation, that the scaling should result in

$$\frac{\partial \bar{u}}{\partial \bar{t}} \text{ and } \frac{\partial^2 \bar{u}}{\partial \bar{x}^2}$$

of the same size (about unity), implies $t_c \alpha / L^2 = 1$ and therefore $t_c = L^2/\alpha$.

Analytical insight. The best way to obtain the scales inherent in a problem is to obtain an exact analytic solution, as we have done in many of the ODE examples in this booklet. However, as a rule this is not possible. Still, often highly simplified analytic solutions can be found for parts of the problem, or for some closely related problem. Such solutions may provide crucial guidance to the nature of the complete solution and to the appropriate scaling of the full problem. We will employ such solutions now to learn about scales in diffusion problems.

One can show that $u = Ae^{-pt}\sin(kx)$ is a solution of (3.36) if $p = \alpha k^2$, for any k. This is the typical solution arising from separation of variables and reflects the dynamics of the space and time in the PDE. Exponential decay in time is a characteristic feature of diffusion processes, and the e-folding time can then be taken as a time scale. This means $t_c = 1/p \sim k^{-2}$. Since k

is related to the spatial wave length λ through $k = 2\pi/\lambda$, it means that t_c depends strongly on the wave length of the sine term $\sin(kx)$. In particular, short waves (as found in noisy signals) with large k decay very rapidly. For the overall solution we are interested in how the longest meaningful wave decays and use that time scale for t_c. The longest wave typically has half a wave length over the domain $[0,L]$: $u = Ae^{-pt}\sin(\pi x/L)$ $(k = \pi/L)$, provided $u(0,t) = u(L,t) = 0$ (with $u_x(L,t) = 0$, the longest wave is $L/4$, but we look at the case with the wave length $L/2$). Then $t_c = L^2/\alpha\pi^{-2}$, but the factor π^{-2} is not important and we simply choose $t_c = L^2/\alpha$, which equals the time scale we arrived at above. We may say that t_c is the time it takes for the diffusion to significantly change the solution in the entire domain.

Another fundamental solution of the diffusion equation is the diffusion of a Gaussian function: $u(x,t) = K(4\pi\alpha t)^{-1/2}\exp\left(-x^2/(4\alpha t)\right)$, for some constant K with the same dimension as u. For the diffusion to be significant at a distance $x = L$, we may demand the exponential factor to have a value of $e^{-1} \approx 0.37$, which implies $t = L^2/(4\alpha)$, but the factor 4 is not of importance, so again, a relevant time scale is $t_c = L^2/\alpha$.

Choosing other scales. The scale u_c is chosen according to the initial condition: $u_c = \max_{x\in(0,L)}|I(x)|$. For a diffusion equation $u_t = \alpha u_{xx}$ with $u = 0$ at the boundaries $x = 0, L$, the solution is bounded by the initial condition $I(x)$. Therefore, the listed choice of u_c implies that $|u| \leq 1$. (The solution $u = Ae^{-pt}\sin(kx)$ is such an example if $k = n\pi/L$ for integer n such that $u = 0$ for $x = 0$ and $x = L$.)

The resulting dimensionless PDE becomes

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{x}^2}, \quad \bar{x} \in (0,1),\ \bar{t} \in (0,\bar{T}], \tag{3.38}$$

with initial condition

$$\bar{u}(\bar{x},0) = \bar{I}(\bar{x}) = \frac{I(x_c\bar{x})}{\max_x |I(x)|}\,.$$

Notice that (3.38) is without physical parameters, but there may be parameters in $I(x)$.

3.2.2 Generalized diffusion PDE

Turning the attention to (3.37), we introduce the dimensionless diffusion coefficient

$$\bar{\alpha}(\bar{x},\bar{y},\bar{z}) = \alpha_c^{-1}\alpha(x_c\bar{x}, y_c\bar{y}, z_c\bar{z}),$$

typically with

$$\alpha_c = \max_{x,y,z} \alpha(x,y,z)\,.$$

The length scales are

$$\bar{x} = \frac{x}{x_c}, \quad \bar{y} = \frac{y}{y_c}, \quad \bar{z} = \frac{z}{z_c}\,.$$

We scale f in a similar fashion:

$$\bar{f}(\bar{x},\bar{y},\bar{z},\bar{t}) = f_c^{-1} f(\bar{x}x_c, \bar{y}y_c\bar{z}z_c, \bar{t}t_c),$$

with

$$f_c = \max_{x,y,z,t} |f(x,y,z,t)|\,.$$

Also assuming that $x_c = y_c = z_c = L$, and $u_c = \max_{x,y,z}(I(x,y,z)$, we end up with the scaled PDE

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \bar{\nabla}\cdot\left(\bar{\alpha}\bar{\nabla}\bar{u}\right) + \beta\bar{f}, \quad \bar{x},\bar{y},\bar{z}\in\bar{\Omega},\ \bar{t}\in(0,\bar{T}]\,. \tag{3.39}$$

Here, $\bar{\nabla}$ means differentiation with respect to dimensionless coordinates $\bar{x}$, $\bar{y}$, and $\bar{z}$. The dimensionless parameter β takes the form

$$\beta = \frac{t_c f_c}{u_c} = \frac{L^2}{\alpha}\frac{\max_{x,y,z,t}|f(x,y,z,t)|}{\max_{x,y,z}|I(x,y,z)|}\,.$$

The scaled initial condition is $\bar{u} = \bar{I}$ as in the 1D case.

An alternative choice of u_c is to make the coefficient $t_c f_c/u_c$ in the source term unity. The scaled PDE now becomes

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \bar{\nabla}\cdot\left(\bar{\alpha}\bar{\nabla}\bar{u}\right) + f, \tag{3.40}$$

but the initial condition features the β parameter:

$$\bar{u}(\bar{x},\bar{y},\bar{z},0) = \frac{I}{t_c f_c} = \beta^{-1}\bar{I}(\bar{x},\bar{y},\bar{z})\,.$$

The β parameter can be interpreted as the ratio of the source term and the terms with u:

$$\beta = \frac{f_c}{u_c/t_c} \sim \frac{|f|}{|u_t|}, \quad \beta = \frac{f_c}{u_c/t_c} = \frac{f_c}{L^2/t_c u_c/L^2} \sim \frac{|f_c|}{|\alpha\nabla^2 u|}\,.$$

We may check that β is really non-dimensional. From the PDE, f must have the same dimensions as $\partial u/\partial t$, i.e., $[\Theta \mathrm{T}^{-1}]$. The dimension of α is more intricate, but from the term αu_{xx} we know that u_{xx} has dimensions $[\Theta \mathrm{L}^{-2}]$, and then α must have dimension $[\mathrm{L}^2\mathrm{T}^{-1}]$ to match the target $[\Theta \mathrm{T}^{-1}]$. In the expression for β we get $[\mathrm{L}^2\Theta\mathrm{T}^{-1}(\mathrm{L}^2\mathrm{T}^{-1}\Theta)^{-1}]$, which equals 1 as it should.

3.2.3 Jump boundary condition

A classical one-dimensional heat conduction problem goes as follows. An insulated rod at some constant temperature U_0 is suddenly heated from one end ($x = 0$), modeled as a constant Dirichlet condition $u(0,t) = U_1 \neq U_0$ at that end. That is, the boundary temperature jumps from U_0 to U_1 at $t = 0$. All the other surfaces of the rod are insulated such that a one-dimensional model is appropriate, but we must explicitly demand $u_x(L,t) = 0$ to incorporate the insulation condition in the one-dimensional model at the end of the domain $x = L$. Heat cannot escape, and since we supply heat at $x = 0$, all of the material will eventually be warmed up to the temperature U_1: $u \to U_1$ as $t \to \infty$.

The initial-boundary value problem reads

$$\varrho c\frac{\partial u}{\partial t} = k\frac{\partial^2 u}{\partial x^2}, \qquad x \in (0,L),\ t \in (0,T], \tag{3.41}$$

$$u(x,0) = U_0, \qquad x \in [0,L], \tag{3.42}$$

$$u(0,t) = U_1, \qquad t \in (0,T], \tag{3.43}$$

$$\frac{\partial}{\partial x}u(L,t) = 0, \qquad t \in (0,T]. \tag{3.44}$$

The PDE (3.41) arises from the energy equation in solids and involves three physical parameters: the density ϱ, the specific heat capacity parameter c,a nd the heat conduction coefficient (from Fourier's law). Dividing by ϱc and introducing $\alpha = k/(\varrho c)$ brings (3.41) on the standard form (3.36). We just use the α parameter in the following.

The natural dimensionless temperature for this problem is

$$\bar{u} = \frac{u - U_0}{U_1 - U_0},$$

since this choice makes $\bar{u} \in [0,1]$. The reason is that u is bounded by the initial and boundary conditions (in the absence of a source term in the PDE), and we have $\bar{u}(\bar{x},0) = 0$, $\bar{u}(\bar{x},\infty) = 1$, and $\bar{u}(0,\bar{t}) = 1$.

The choice of t_c is as in the previous cases. We arrive at the dimensionless initial-boundary value problem

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{x}^2}, \qquad \bar{x} \in (0,1),\ \bar{t} \in (0,\bar{T}], \tag{3.45}$$

$$\bar{u}(\bar{x},0) = 0, \qquad \bar{x} \in [0,1], \tag{3.46}$$

$$\bar{u}(0,\bar{t}) = 1, \qquad \bar{t} \in (0,\bar{T}], \tag{3.47}$$

$$\frac{\partial}{\partial \bar{x}}\bar{u}(1,\bar{t}) = 0, \qquad \bar{t} \in (0,\bar{T}]. \tag{3.48}$$

The striking feature is that there are no physical parameters left in this problem. One simulation can be carried out for $\bar{u}(\bar{x},\bar{t})$, see Figure 3.3, and the temperature in a rod of any material and any constant initial and boundary temperature can be retrieved by

$$u(x,t) = U_0 + (U_1 - U_0)\bar{u}(x/L, t\alpha/L^2)\,.$$

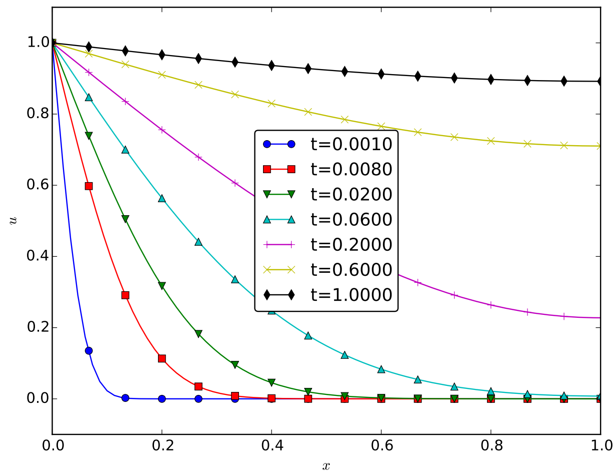


Fig. 3.3 Scaled temperature in an isolated rod suddenly heated from the end.

3.2.4 Oscillating Dirichlet condition

Now we address a heat equation problem where the temperature is oscillating on the boundary $x = 0$:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \qquad x \in (0,L),\ t \in (0,T], \tag{3.49}$$

$$u(x,0) = U_0, \qquad x \in [0,L], \tag{3.50}$$

$$u(0,t) = U_0 + A\sin(\omega t), \qquad t \in (0,T], \tag{3.51}$$

$$\frac{\partial}{\partial x} u(L,t) = 0, \qquad t \in (0,T]. \tag{3.52}$$

One important physical application is temperature oscillations in the ground, either day and night variations at a short temporal and spatial scale, or seasonal variations in the Earth's crust. An important modeling assumption is (3.52), which means that the boundary $x = L$ is placed sufficiently far from $x = 0$ such that the solution is much damped and basically constant so $u_x = 0$ is a reasonable condition.

Scaling issues. Since the boundary temperature is oscillating around the initial condition, we expect $u \in [U_0 - A, U_0 + A]$. The dimensionless temperature is therefore taken as

$$\bar{u} = \frac{u - U_0}{A},$$

such that $\bar{u} \in [-1, 1]$.

What is an appropriate time scale? There will be two time scales involved, the oscillations $\sin(\omega t)$ with period $P = 2\pi/\omega$ at the boundary and the "speed of diffusion", or more specifically the "speed of heat conduction" in the present context, where $t_c = x_c^2/\alpha$ is the appropriate scale, x_c being the length scale. Choosing the right length scale is not obvious. As we shall see, the standard choice $x_c = L$ is not a good candidate, but to understand why, we need to examine the solution, either through simulations or through a closed-form formula. We are so lucky in this relatively simple pedagogical problem that one can find an exact solution of a related problem.

Exact solution. As usual, investigating the exact solution of the model problem can illuminate the involved scales. For this particular initial-boundary value problem the exact solution as $t \to \infty$ (such that the initial condition $u(x, 0) = U_0$ is forgotten) and $L \to \infty$ (such that (3.52) is certainly valid) can be shown to be

$$u(x, t) = U_0 - Ae^{-bx} \sin(bx - \omega t), \quad b = \sqrt{\frac{\omega}{2\alpha}}. \tag{3.53}$$

This solution is of the form $e^{-bx} g(x - ct)$, i.e., a damped wave that moves to the right with velocity c and a damped amplitude e^{-bx}. This is perhaps more easily seen if we make a rewrite

$$u(x, t) = U_0 - Ae^{-bx} \sin\left(b(x - ct)\right), \quad c = \omega/b = \sqrt{2\alpha\omega},\ b = \sqrt{\frac{\omega}{2\alpha}}.$$

Time and length scales. The boundary oscillations lead to the time scale $t_c = 1/\omega$. The speed of the wave suggests another time scale: the time it takes to propagate through the domain, which is L/c, and hence $t_c = L/c = L/\sqrt{2\alpha\omega}$.

One can argue that L is not the appropriate length scale, because u is damped by e^{-bx}. So, for $x > 4/b$, u is close to zero. We may instead use

$1/b$ as length scale, which is the e-folding distance of the damping factor, and base t_c on the time it takes a signal to propagate one length scale, $t_c^{-1} = bc = \omega$. Similarly, the time scale based on the "speed of diffusion" changes to $t_c^{-1} = b^2\alpha = \frac{1}{2}\omega$ if we employ $1/b$ as length scale.

To summarize, we have three candidates for the time scale: $t_c = L^2/\alpha$ (diffusion through the entire domain), $t_c = 2/\omega$ (diffusion through a distance $1/b$ where u is significantly different from zero), and $t_c = 1/\omega$ (wave movement over a distance $1/b$).

Let us look at the dimensionless exact solution to see if it can help with the choice of scales. We introduce the dimensionless parameters

$$\beta = bx_c = x_c\sqrt{\frac{\omega}{2\alpha}}, \quad \gamma = \omega t_c\,.$$

The scaled solution becomes

$$\bar{u}(\bar{x},\bar{t};\beta,\gamma) = e^{-\beta\bar{x}}\sin(\gamma\bar{t} - \beta\bar{x})\,.$$

The three choices of γ, implied by the three choices of t_c, are

$$\gamma = \begin{cases} 1, & t_c = 1/\omega, \\ 2, & t_c = 2/\omega, \\ 2\beta^2, & t_c = L^2/\alpha,\ x_c = L \end{cases} \tag{3.54}$$

The former two choices leave only β as parameter in $\bar{u}$, and with $x_c = 1/b$ as length scale, β becomes unity, and there are no parameters in the dimensionless solution:

$$\bar{u}(\bar{x},\bar{t}) = e^{-\bar{x}}\sin(\bar{t} - \bar{x})\,. \tag{3.55}$$

Therefore, $x_c = 1/b$ and $t_c = 1/\omega$ (or $t_c = 2/\omega$, but the factor 2 is of no importance) are the most appropriate scales.

To further argue why (3.55) demonstrates that these scales are preferred, think of ω as large. Then the wave is damped over a short distance and there will be a thin boundary layer of temperature oscillations near $x = 0$ and little changes in u in the rest of the domain. The scaling (3.55) resolves this problem by using $1/b \sim \omega^{-1/2}$ as length scale, because then the boundary layer thickness is independent of ω. The length of the domain can be chosen as, e.g., $4/b$ such that $\bar{u} \approx 0$ at the end $x = L$. The length scale $1/b$ helps us to zoom in on the part of u where significant changes take place.

In the other limit, ω small, b becomes small, and the wave is hardly damped in the domain $[0, L]$ unless L is large enough. The imposed boundary condition on $x = L$ in fact requires u to be approximately constant so its derivative vanishes, and this property can only be obtained if L is large enough to ensure that the wave becomes significantly damped. Therefore, the length scale is dictated by b, not L, and L should be adapted to b, typically $L \geq 4/b$ if $e^{-4} \approx 0.018$ is considered enough damping to consider $\bar{u} \approx 0$ for the boundary condition. This means that $x \in [0, 4/b]$ and then $\bar{x} \in [0, 4]$. Increasing the spatial domain to $[0, 6]$ implies a damping $e^{-6} \approx 0.0025$, if more accuracy is desired in the boundary condition.

The scaled problem. Based on the discussion of scales above, we arrive at the following scaled initial-boundary value problem:

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{1}{2}\frac{\partial^2 \bar{u}}{\partial x^2}, \qquad \bar{x} \in (0,4),\ \bar{t} \in (0,\bar{T}], \tag{3.56}$$

$$\bar{u}(\bar{x},0) = 0, \qquad \bar{x} \in [0,1], \tag{3.57}$$

$$\bar{u}(0,\bar{t}) = \sin(\bar{t}), \qquad \bar{t} \in (0,\bar{T}], \tag{3.58}$$

$$\frac{\partial}{\partial \bar{x}}\bar{u}(\bar{L},\bar{t}) = 0, \qquad \bar{t} \in (0,\bar{T}]. \tag{3.59}$$

The coefficient in front of the second-derivative is $\frac{1}{2}$ because

$$\frac{t_c\alpha}{1/b^2} = \frac{b^2\alpha}{\omega} = \frac{1}{2}\,.$$

We may, of course, choose $t_c = 2/\omega$ and get rid of the $\frac{1}{2}$ factor, if desired, but then it turns up in (3.58) instead, as $\sin(2\bar{t})$.

The boundary condition at $\bar{x} = \bar{L}$ is only an approximation and relies on sufficient damping of $\bar{u}$ to consider it constant ($\partial/\partial\bar{x} = 0$) in space. We could, therefore, assign the condition $\bar{u} = 0$ instead at $\bar{x} = \bar{L}$.

Simulations. The file `session.py` contains a function `solver_diffusion_FE` for solving a diffusion equation in one dimension. This function can be used to solve the system (3.56)-(3.59), see `diffusion_oscillatory_BC`.

Movie 3: Diffusion wave. `https://github.com/hplgit/scaling-book/raw/master/doc/pub/book/html/mov-scaling/diffusion_osc_BC/movie.mp4`

3.3 Reaction-diffusion equations

3.3.1 Fisher's equation

Fisher's equation is essentially the logistic equation at each point for population dynamics (see Section 2.1.9) combined with spatial movement through ordinary diffusion:

$$\frac{\partial u}{\partial t} = \alpha\frac{\partial^2 u}{\partial x^2} + \varrho u(1 - u/M)\,. \tag{3.60}$$

This PDE is also known as the KPP equation after Kolmogorov, Petrovsky, and Piskynov (who introduced the equation independently of Fisher).

Setting

$$\bar{x} = \frac{x}{x_c}, \quad \bar{t} = \frac{t}{t_c}, \quad \bar{u} = \frac{u}{u_c},$$

results in

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{t_c\alpha}{x_c^2}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + t_c\varrho\bar{u}(1 - u_c\bar{u}/M)\,.$$

Balance of all terms. If all terms are equally important, the scales can be determined from demanding the coefficients to be unity. Reasoning as for the logistic ODE in Section 2.1.9, we may choose $t_c = 1/\varrho$. Then the coefficient in the diffusion term dictates the length scale $x_c = \sqrt{t_c\alpha}$. A natural scale for u is M, since M is the upper limit of u in the model (cf. the logistic term). Summarizing,

$$u_c = M, \quad t_c = \frac{1}{\varrho}, \quad x_c = \sqrt{\frac{\alpha}{\varrho}},$$

and the scaled PDE becomes

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \bar{u}(1 - \bar{u})\,. \tag{3.61}$$

With this scaling, the length scale $x_c = \sqrt{\alpha/\varrho}$ is not related to the domain size, so the scale is particularly relevant for infinite domains.

An open question is whether the time scale should be based on the diffusion process rather than the initial exponential growth in the logistic term. The diffusion time scale means $t_c = x_c^2/\alpha$, but demanding the logistic term then to have a unit coefficient forces $x_c^2\varrho/\alpha = 1$, which implies $x_c = \sqrt{\alpha/\varrho}$ and $t_c = 1/\varrho$. That is, equal balance of the three terms gives a unique choice of the time and length scale.

Fixed length scale. Assume now that we fix the length scale to be L, either the domain size or some other naturally given length. With $x_c = L$, $t_c = \varrho^{-1}$, $u_c = M$, we get

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \beta\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \bar{u}(1 - \bar{u}), \tag{3.62}$$

where β is a dimensionless number

$$\beta = \frac{\alpha}{\varrho L^2} = \frac{\varrho^{-1}}{L^2/\alpha}\,.$$

The last equality demonstrates that β measures the ratio of the time scale for exponential growth in the beginning of the logistic process and the time scale of diffusion L^2/α (i.e., the time it takes to transport a signal by diffusion through the domain). For small β we can neglect the diffusion and spatial movements, and the PDE is essentially a logistic ODE at each point, while for large β, diffusion dominates, and t_c should in that case be based on the diffusion time scale L^2/α. This leads to the scaled PDE

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \beta^{-1}\bar{u}(1-\bar{u}), \tag{3.63}$$

showing that a large β encourages omission of the logistic term, because the point-wise growth takes place over long time intervals while diffusion is rapid. The effect of diffusion is then more prominent and it suffices to solve $\bar{u}_{\bar{t}} = \bar{u}_{\bar{x}\bar{x}}$. The observant reader will in this latter case notice that $u_c = M$ is an irrelevant scale for u, since logistic growth with its limit is not of importance, so we implicitly assume that another scale u_c has been used, but that scale cancels anyway in the simplified PDE $\bar{u}_{\bar{t}} = \bar{u}_{\bar{x}\bar{x}}$.

3.3.2 Nonlinear reaction-diffusion PDE

A general, nonlinear reaction-diffusion equation in 1D looks like

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} + f(u). \tag{3.64}$$

By scaling the nonlinear reaction term $f(u)$ as $f_c \bar{f}(u_c\bar{u})$, where f_c is a characteristic size of $f(u)$, typically the maximum value, one gets a non-dimensional PDE like

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{t_c\alpha}{x_c^2}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \frac{t_c f_c}{u_c}\bar{f}(u_c\bar{u}).$$

The characteristic size of u can often be derived from boundary or initial conditions, so we first assume that u_c is given. This fact uniquely determines the space and time scales by demanding that all three terms are equally important and of unit size:

$$t_c = \frac{u_c}{f_c}, \quad x_c = \sqrt{\frac{\alpha u_c}{f_c}}.$$

The corresponding PDE reads

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \bar{f}(u_c\bar{u}). \tag{3.65}$$

If x_c is based on some known length scale L, balance of all three terms can be used to determine u_c and t_c:

$$t_c = \frac{L^2}{\alpha}, \quad u_c = \frac{L^2 f_c}{\alpha}.$$

This scaling only works if f is nonlinear, otherwise u_c cancels and there is no freedom to constrain this scale.

With given L and u_c, there are two choices of t_c since it can be based on the diffusion or the reaction time scales. With the reaction scale, $t_c = u_c/f_c$, one arrives a the PDE

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \beta \frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \bar{f}(u_c \bar{u}), \tag{3.66}$$

where

$$\beta = \frac{\alpha u_c}{L^2 f_c} = \frac{u_c/f_c}{L^2/\alpha}$$

is a dimensionless number reflecting the ratio of the reaction time scale and the diffusion time scale. On the contrary, with the diffusion time scale, $t_c = L^2/\alpha$, the scaled PDE becomes

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \beta^{-1} \bar{f}(u_c \bar{u})\,. \tag{3.67}$$

The size of β in an application will determine which of the scalings that is most appropriate.

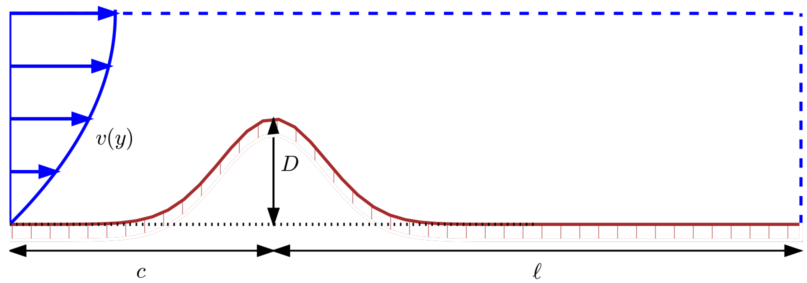
3.4 The convection-diffusion equation

3.4.1 Convection-diffusion without a force term

We now add a convection term $\boldsymbol{v} \cdot \nabla u$ to the diffusion equation to obtain the well-known convection-diffusion equation:

$$\frac{\partial u}{\partial t} + \boldsymbol{v} \cdot \nabla u = \alpha \nabla^2 u, \quad x,y,z \in \Omega,\ t \in (0,T]\,. \tag{3.68}$$

The velocity field $\boldsymbol{v}$ is prescribed, and its characteristic size V is normally clear from the problem description. In the sketch below, we have some given flow over a bump, and u may be the concentration of some substance in the fluid. Here, V is typically $\max_y v(y)$. The characteristic length L could be the entire domain, $L = c + \ell$, or the height of the bump, $L = D$. (The latter is the important length scale for the flow.)



Inserting

$$\bar{x} = \frac{x}{x_c},\ \bar{y} = \frac{y}{y_c},\ \bar{z} = \frac{z}{z_c},\ \bar{t} = \frac{t}{t_c},\ \bar{\boldsymbol{v}} = \frac{\boldsymbol{v}}{V},\ \bar{u} = \frac{u}{u_c}$$

in (3.68) yields

$$\frac{u_c}{t_c}\frac{\partial \bar{u}}{\partial \bar{t}} + \frac{u_c V}{L}\bar{\boldsymbol{v}}\cdot\bar{\nabla}\bar{u} = \frac{\alpha u_c}{L^2}\bar{\nabla}^2\bar{u}, \quad \bar{x},\bar{y},\bar{z}\in\Omega,\ \bar{t}\in(0,\bar{T}]\,.$$

For u_c we simply introduce the symbol U, which we may estimate from an initial condition. It is not critical here, since it vanishes from the scaled equation anyway, as long as there is no source term present. With some velocity measure V and length measure L, it is tempting to just let $t_c = L/V$. This is the characteristic time it takes to transport a signal by convection through the domain. The alternative is to use the diffusion length scale $t_c = L^2/\alpha$. A common physical scenario in convection-diffusion problems is that the convection term $\boldsymbol{v}\cdot\nabla u$ dominates over the diffusion term $\alpha\nabla^2 u$. Therefore, the time scale for convection (L/V) is most appropriate of the two. Only when the diffusion term is very much larger than the convection term (corresponding to very small Peclet numbers, see below) $t_c = L^2/\alpha$ is the right time scale.

The non-dimensional form of the PDE with $t_c = L/V$ becomes

$$\frac{\partial \bar{u}}{\partial \bar{t}} + \bar{\boldsymbol{v}}\cdot\bar{\nabla}\bar{u} = \mathrm{Pe}^{-1}\bar{\nabla}^2\bar{u}, \quad \bar{x},\bar{y},\bar{z}\in\Omega,\ \bar{t}\in(0,\bar{T}], \tag{3.69}$$

where Pe is the *Peclet number*,

$$\mathrm{Pe} = \frac{LV}{\alpha}\,.$$

Estimating the size of the convection term $\boldsymbol{v}\cdot\nabla u$ as VU/L and the diffusion term $\alpha\nabla^2 u$ as $\alpha U/L^2$, we see that the Peclet number measures the ratio of the convection and the diffusion terms:

$$\mathrm{Pe} = \frac{\text{convection}}{\text{diffusion}} = \frac{VU/L}{\alpha U/L^2} = \frac{LV}{\alpha}\,.$$

In case we use the diffusion time scale $t_c = L^2/\alpha$, we get the non-dimensional PDE

$$\frac{\partial \bar{u}}{\partial \bar{t}} + \mathrm{Pe}\,\bar{\boldsymbol{v}}\cdot\bar{\nabla}\bar{u} = \bar{\nabla}^2\bar{u}, \quad \bar{x},\bar{y},\bar{z}\in\Omega,\ \bar{t}\in(0,\bar{T}]\,. \tag{3.70}$$

Discussion of scales and balance of terms in the PDE

We see that (3.69) and (3.70) are not equal, and they are based on two different time scales. For moderate Peclet numbers around 1, all terms have the same size in (3.69), i.e., a size around unity. For large Peclet numbers, (3.69) expresses a balance between the time derivative term and the convection term, both of size unity, and then there is a very small $\mathrm{Pe}^{-1}\bar{\nabla}^2\bar{u}$ term because Pe is large and $\bar{\nabla}^2\bar{u}$ should be of size unity. That the convection term dominates over the diffusion term is consistent with the time scale $t_c = L/V$ based on convection transport. In this case, we can neglect the diffusion term as Pe goes to infinity and work with a pure convection (or advection) equation

$$\frac{\partial \bar{u}}{\partial \bar{t}} + \bar{\boldsymbol{v}}\cdot\bar{\nabla}\bar{u} = 0\,.$$

For small Peclet numbers, $\mathrm{Pe}^{-1}\bar{\nabla}^2\bar{u}$ becomes very large and can only be balanced by two terms that are supposed to be unity of size. The time-derivative and/or the convection term must be much larger than unity, but that means we use suboptimal scales, since right scales imply that $\partial\bar{u}/\partial\bar{t}$ and $\bar{v}\cdot\bar{\nabla}\bar{u}$ are of order unity. Switching to a time scale based on diffusion as the dominating physical effect gives (3.70). For very small Peclet numbers this equation tells that the time-derivative balances the diffusion. The convection term $\bar{\boldsymbol{v}}\cdot\bar{\nabla}\bar{\boldsymbol{u}}$ is around unity in size, but multiplied by a very small coefficient Pe, so this term is negligible in the PDE. An approximate PDE for small Peclet numbers is therefore

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \bar{\nabla}^2\bar{u}\,.$$

Scaling can, with the above type of reasoning, be used to neglect terms from a differential equation under precise mathematical conditions.

3.4.2 Stationary PDE

Suppose the problem is stationary and that there is no need for any time scale. How is this type of convection-diffusion problem scaled? We get

$$\frac{VU}{L}\bar{\boldsymbol{v}}\cdot\bar{\nabla}\bar{u} = \frac{\alpha U}{L^2}\bar{\nabla}^2\bar{u},$$

or

$$\bar{\boldsymbol{v}}\cdot\bar{\nabla}\bar{u} = \mathrm{Pe}^{-1}\bar{\nabla}^2\bar{u}\,. \tag{3.71}$$

This scaling only "works" for moderate Peclet numbers. For very small or very large Pe, either the convection term $\bar{\boldsymbol{v}}\cdot\bar{\nabla}\bar{u}$ or the diffusion term $\bar{\nabla}^2\bar{u}$ must deviate significantly from unity.

Consider the following 1D example to illustrate the point: $\boldsymbol{v} = v\boldsymbol{i}$, $v > 0$ constant, a domain $[0, L]$, with boundary conditions $u(0) = 0$ and $u(L) = U_L$. (The vector $\boldsymbol{i}$ is a unit vector in x direction.) The problem with dimensions is now

$$vu' = \alpha u'', \quad u(0) = 0,\ u(L) = U_L\,.$$

Scaling results in

$$\frac{d\bar{u}}{d\bar{x}} = \mathrm{Pe}^{-1}\frac{d^2\bar{u}}{d\bar{x}^2}, \quad \bar{x}\in(0,1), \quad \bar{u}(0) = 0,\ \bar{u}(1) = 1,$$

if we choose $U = U_L$. The solution of the scaled problem is

$$\bar{u}(\bar{x}) = \frac{1-e^{\bar{x}\mathrm{Pe}}}{1-e^{\mathrm{Pe}}}\,.$$

Figure 3.4 indicates how $\bar{u}$ depends on Pe: small Pe values give approximately a straight line while large Pe values lead to a *boundary layer* close to $x = 1$, where the solution changes very rapidly.

We realize that for large Pe,

$$\max_{\bar{x}}\frac{d\bar{u}}{d\bar{x}} \approx \mathrm{Pe}, \quad \max_{\bar{x}}\frac{d^2\bar{u}}{d\bar{x}^2} \approx \mathrm{Pe}^2,$$

which are consistent results with the PDE, since the double derivative term is multiplied by Pe^{-1}. For small Pe,

$$\max_{\bar{x}}\frac{d\bar{u}}{d\bar{x}} \approx 1, \quad \max_{\bar{x}}\frac{d^2\bar{u}}{d\bar{x}^2} \approx 0,$$

which is also consistent with the PDE, since an almost vanishing second-order derivative is multiplied by a very large coefficient Pe^{-1}. However, we have a problem with very large derivatives of $\bar{u}$ when Pe is large.

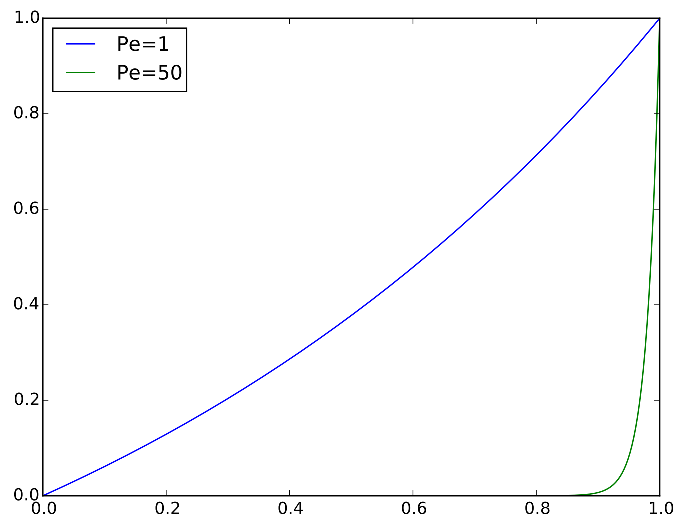


Fig. 3.4 Solution of scaled problem for 1D convection-diffusion.

To arrive at a proper scaling for large Peclet numbers, we need to remove the Pe coefficient from the differential equation. There are only two scales at our disposals: u_c and x_c for u and x, respectively. The natural value for u_c is the boundary value U_L at $x = L$. The scaling of $Vu_x = \alpha u_{xx}$ then results in

$$\frac{d\bar{u}}{d\bar{x}} = \frac{\alpha}{Vx_c}\frac{d^2\bar{u}}{d\bar{x}^2}, \quad \bar{x} \in (0, \bar{L}), \quad \bar{u}(0) = 0,\ \bar{u}(\bar{L}) = 1,$$

where $\bar{L} = L/x_c$. Choosing the coefficient $\alpha/(Vx_c)$ to be unity results in the scale $x_c = \alpha/V$, and $\bar{L}$ becomes Pe. The final, scaled boundary-value problem is now

$$\frac{d\bar{u}}{d\bar{x}} = \frac{d^2\bar{u}}{d\bar{x}^2}, \quad \bar{x} \in (0, \mathrm{Pe}), \quad \bar{u}(0) = 0,\ \bar{u}(\mathrm{Pe}) = 1,$$

with solution

$$\bar{u}(\bar{x}) = \frac{1 - e^{\bar{x}}}{1 - e^{\mathrm{Pe}}}\,.$$

Figure 3.5 displays $\bar{u}$ for some Peclet numbers, and we see that the shape of the graphs are the same with this scaling. For large Peclet numbers we realize that $\bar{u}$ and its derivatives are around unity ($1 - e^{\mathrm{Pe}} \approx -e^{\mathrm{Pe}}$), but for small Peclet numbers $d\bar{u}/d\bar{x} \sim \mathrm{Pe}^{-1}$.

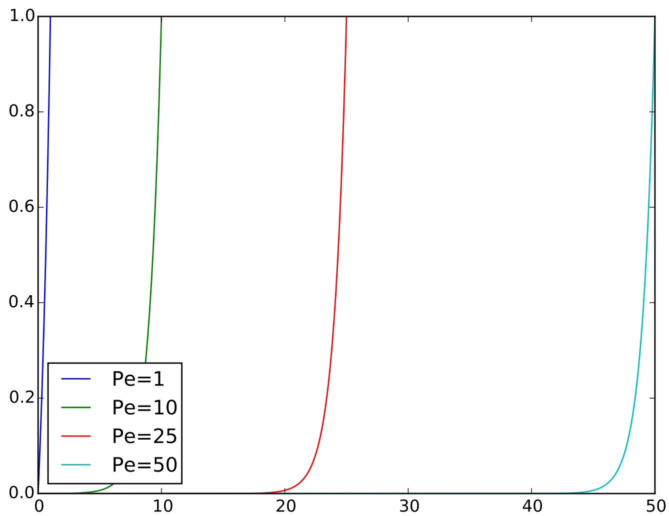


Fig. 3.5 Solution of scaled problem where the length scale depends on the Peclet number.

The conclusion is that for small Peclet numbers, $x_c = L$ is an appropriate length scale. The scaled equation $\mathrm{Pe}\,\bar{u}' = \bar{u}''$ indicates that $\bar{u}'' \approx 0$, and the solution is close to a straight line. For large Pe values, $x_c = \alpha/V$ is an appropriate length scale, and the scaled equation $\bar{u}' = \bar{u}''$ expresses that the terms $\bar{u}'$ and $\bar{u}''$ are equal and of size around unity.

3.4.3 Convection-diffusion with a source term

Let us add a force term $f(\boldsymbol{x},t)$ to the convection-diffusion equation:

$$\frac{\partial u}{\partial t} + \boldsymbol{v}\cdot\nabla u = \alpha\nabla^2 u + f\,. \tag{3.72}$$

The scaled version reads

$$\frac{\partial \bar{u}}{\partial \bar{t}} + \frac{t_c V}{L}\bar{\boldsymbol{v}}\cdot\bar{\nabla}\bar{u} = \frac{t_c\alpha}{L^2}\bar{\nabla}^2\bar{u} + \frac{t_c f_c}{u_c}\bar{f}\,.$$

We can base t_c on convective transport: $t_c = L/V$. Now, u_c could be chosen to make the coefficient in the source term unity: $u_c = t_c f_c = L f_c/V$. This leaves us with

$$\frac{\partial \bar{u}}{\partial \bar{t}} + \bar{\boldsymbol{v}} \cdot \bar{\nabla} \bar{u} = \mathrm{Pe}^{-1} \bar{\nabla}^2 \bar{u} + \bar{f}\,.$$

In the diffusion limit, we base t_c on the diffusion time scale: $t_c = L^2/\alpha$, and the coefficient of the source term set to unity determines u_c according to

$$\frac{L^2 f_c}{\alpha u_c} = 1 \quad \Rightarrow \quad u_c = \frac{L^2 f_c}{\alpha}\,.$$

The corresponding PDE reads

$$\frac{\partial \bar{u}}{\partial \bar{t}} + \mathrm{Pe}\, \bar{\boldsymbol{v}} \cdot \bar{\nabla} \bar{u} = \bar{\nabla}^2 \bar{u} + \bar{f},$$

so for small Peclet numbers, which we have, the convective term can be neglected and we get a pure diffusion equation with a source term.

What if the problem is stationary? Then there is no time scale and we get

$$\frac{V u_c}{L} \bar{\boldsymbol{v}} \cdot \bar{\nabla} \bar{u} = \frac{u_c \alpha}{L^2} \bar{\nabla}^2 \bar{u} + f_c \bar{f},$$

or

$$\bar{\boldsymbol{v}} \cdot \bar{\nabla} \bar{u} = \mathrm{Pe}^{-1} \bar{\nabla}^2 \bar{u} + \frac{f_c L}{V u_c} \bar{f}\,.$$

Again, choosing u_c such that the source term coefficient is unity leads to $u_c = f_c L/V$. Alternatively, u_c can be based on the initial condition, with similar results as found in the sections on the wave and diffusion PDEs.

3.5 Exercises

Problem 3.1: Stationary Couette flow

A fluid flows between two flat plates, with one plate at rest while the other moves with velocity U_0. This classical flow case is known as stationary Couette flow.

a) Directing the x axis in the flow direction and letting y be a coordinate perpendicular to the walls, one can assume that the velocity field simplifies to $\boldsymbol{u} = u(y)\boldsymbol{i}$. Show from the Navier-Stokes equations that the boundary-value problem for $u(y)$ is

$$u''(u) = 0, \quad u(0) = 0,\ u(H) = U_0\,.$$

We have here assumed at $y = 0$ corresponds to the plate at rest and that $y = H$ represents the plate that moves. There are no pressure gradients present in the flow.

b) Scale the problem in a) and show that the result has no physical parameters left in the model:

$$\frac{d^2\bar{u}}{d\bar{y}^2} = 0, \quad \bar{u}(0) = 0, \ \bar{u}(1) = 1\,.$$

c) We can compute $\bar{u}(\bar{y})$ from one numerical simulation (or a straightforward integration of the differential equation). Set up the formula that finds $u(y; H, u_0)$ from $\bar{u}(\bar{y})$ for any values of H and U_0.
Filename: `stationary_Couette`.

Remarks. The problem for u is a classical two-point boundary-value problem in applied mathematics and arises in a number of applications, where Couette flow is just one example. Heat conduction is another example: u is temperature, and the heat conduction equation for an insulated rod reduces to $u'' = 0$ under stationary conditions and no heat source. Controlling the end $x = 0$ at 0 degrees Celsius the other end $x = L$ at U_0 degrees Celsius, gives the same boundary conditions as in the above flow problem. The scaled problem is of course the same whether we have flow of fluid or heat.

Exercise 3.2: Couette-Poiseuille flow

Viscous fluid flow between two infinite flat plates $z = 0$ and $z = H$ is governed by

$$\mu u''(z) = -\beta \tag{3.73}$$
$$u(0) = 0, \tag{3.74}$$
$$u(H) = U_0\,. \tag{3.75}$$

Here, $u(z)$ is the fluid velocity in x direction (perpendicular to the z axis), μ is the dynamic viscosity of the fluid, β is a positive constant pressure gradient, and U_0 is the constant velocity of the upper plate $z = H$ in x direction. The model represents Couette flow for $\beta = 0$ and Poiseuille flow for $U_0 = 0$.

a) Find the exact solution $u(z)$. Point out how β and U_0 influence the magnitude of u.

Solution. SymPy can integrate the differential equation twice and fit the integration constants to the boundary conditions:

```
import sympy as sym
mu, beta, z, H = sym.symbols('mu beta z H',
                             real=True, positive=True)
U0, C1, C2 = sym.symbols('U0 C1 C2', real=True)
```

```
# Integrate u''(z) = -beta/mu twice and add integration constants
u = sym.integrate(sym.integrate(-beta/mu, z) + C1, z) + C2

# Use the boundary conditions
eq = [sym.Eq(u.subs(z, 0), 0),
      sym.Eq(u.subs(z, H), U0)]
s = sym.solve(eq, [C1, C2])
print s
u = u.subs(C1, s[C1]).subs(C2, s[C2])
u = sym.simplify(sym.expand(u))
```

The result becomes

$$u(z) = \frac{z}{2H\mu}\left(H\beta(H-z) + 2U_0\mu\right).$$

The maximum value of u is found by

```
# Find max u
dudz = sym.diff(u, z)
s = sym.solve(dudz, z)
print s
umax = u.subs(z, s[0])
umax = sym.simplify(sym.expand(umax))
```

and reads

$$\max_z u = \frac{H^2\beta}{8\mu} + \frac{U_0}{2} + \frac{U_0^2\mu}{2H^2\beta}.$$

If the pressure gradient is the dominating driving force, we can neglect the U_0 terms: $\max_z u = H^2\beta/(8\mu)$. In case the movement of the upper plate is much more important than the pressure gradient for driving the flow, we can neglect the β terms. However, we must then resort to the $u(z)$ expression for $\beta = 0$, $u(z) = zU_0/H$, and realize that the maximum then is obtained at the boundary for $z = H$: $\max_z h = U_0$ (as intuitively obvious).

b) Scale the problem.

Solution. Introducing

$$\bar{z} = \frac{z}{z_c}, \quad \bar{u}(\bar{z}) = \frac{u(z_c\bar{z})}{u_c},$$

in the equation gives

$$\frac{d^2\bar{u}}{d\bar{z}^2} = -\frac{z_c^2\beta}{\mu u_c}.$$

The natural scale for z_c is H since that makes $\bar{z} \in [0,1]$. For the two terms in the differential equation to be of order unity (with a correct scaling, the left-hand side should be of order unity), we must have

$$u_c = \frac{H^2\beta}{\mu}\,.$$

The boundary value problem is

$$\begin{aligned}
\frac{d^2\bar{u}}{d\bar{z}^2} &= -1, && \bar{z} \in (0,1),\\
\bar{u}(0) &= 0,\\
\bar{u}(1) &= \alpha,
\end{aligned}$$

where α is a dimensionless number

$$\alpha = \frac{\mu U_0}{H^2\beta}\,.$$

This is meaningful only for $\beta \neq 0$.

Looking at the exact solution, we see that $\max_z u = H^2\beta/(8\mu)$, and with this $\max_z u$ as u_c we get a differential equation $\bar{u}'' = -8$ instead, and $\bar{u} \in [0,1]$ (if $U_0 = 0$). However, the factor 1 or 8 on the right-hand side is not significant, neither if $\bar{u} \in [0,1]$ or $\bar{u} \in [0,8]$.

The scale u_c used above is relevant if the pressure gradient is the dominating force. If U_0 is more important than β, or $\beta = 0$, we choose $u_c = U_0$ and get instead

$$\frac{d^2\bar{u}}{d\bar{z}^2} = -\alpha^{-1}, \qquad \bar{z} \in (0,1), \tag{3.76}$$

$$\bar{u}(0) = 0, \tag{3.77}$$

$$\bar{u}(1) = 1\,. \tag{3.78}$$

Filename: `Couette_wpressure`.

Exercise 3.3: Pulsatile pipeflow

The flow of a viscous fluid in a straight pipe with circular cross section with radius R is governed by

$$\varrho\frac{\partial u}{\partial t} = \frac{\mu}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) - P(t), \qquad r\in(0,R),\ t\in(0,T], \tag{3.79}$$

$$\frac{\partial u}{\partial r}(0,t) = 0, \qquad t\in(0,T], \tag{3.80}$$

$$u(R,t) = 0, \qquad t\in(0,T], \tag{3.81}$$

$$u(r,0) = 0, \qquad r\in[0,R]. \tag{3.82}$$

The quantity $u(r,t)$ is the fluid velocity, $P(t)$ is a given pressure gradient, ϱ is the fluid density, and μ is the dynamic viscosity.

Assume $P(t) = A\cos\omega t$. Scale the problem and identify appropriate dimensionless numbers. Thereafter, assume $P(t)$ is a more complicated function, but still period with period p. Discuss how the scaling can be extended to this case.

Solution. We introduce dimensionless quantities:

$$\bar r = \frac{r}{R},\quad \bar t = \frac{t}{t_c},\quad \bar u = \frac{u}{u_c}\,.$$

The function $P(t)$ can be scaled as

$$\bar P(\bar t) = \frac{P(t_c\bar t)}{A} = \sin(t_c\omega\bar t)\,.$$

Inserted in the PDE, we get

$$\frac{\partial\bar u}{\partial\bar t} = \frac{t_c\nu}{R^2}\frac{1}{\bar r}\frac{\partial}{\partial\bar r}\left(\bar r\frac{\partial\bar u}{\partial\bar r}\right) - \frac{t_cA}{\varrho u_c}\sin(t_c\omega\bar t),$$

where $\mu = \mu/\varrho$.

The scale for u can be explored by seeking an analytical solution of the problem. Such solutions do exist, they are typically series expansions of Bessel functions, and it is not so easy to extract a simple expression for the maximum value of $|u(r,t)|$. A simpler approach is to estimate u_c by demanding the coefficient in the pressure term to be of unit size:

$$u_c = \frac{t_cA}{\varrho}\,.$$

There are two choices of time scales: the pressure time scale $t_c = 1/\omega$ and the viscosity (or diffusion) time scale $t_c = R^2/nu$. With the latter, we get $u_c = R^2A/\mu$ and

$$\frac{\partial\bar u}{\partial\bar t} = \frac{1}{\bar r}\frac{\partial}{\partial\bar r}\left(\bar r\frac{\partial\bar u}{\partial\bar r}\right) - \sin(\alpha\bar t),$$

where

$$\alpha = \frac{R^2\omega}{\nu} = \frac{R^2/\nu}{1/\omega},$$

showing that α is the ratio of the viscosity time scale and the pressure oscillation time scale.

With the pressure time scale we have

$$u_c = \frac{\varrho A}{\omega},$$

and the scaled PDE becomes

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \alpha^{-1}\frac{1}{\bar{r}}\frac{\partial}{\partial \bar{r}}\left(\bar{r}\frac{\partial \bar{u}}{\partial \bar{r}}\right) - \sin(\bar{t})\,.$$

In both cases the scaled boundary conditions become

$$\frac{\partial \bar{u}}{\partial \bar{t}} = 0, \quad \bar{u}(1,\bar{t}) = 0,$$

for $t \in (0, T/t_c]$, and $\bar{u}(\bar{r},0) = 0$ for $\bar{r} \in [0,1]$.

If $P(t)$ is not sinusoidal but periodic with period p, we have that P is a function of ωt as above, with $\omega = 2\pi/p$. Everything in the scaling remains the same, just the sin term changes to $P(\alpha\bar{t})$ if the time scale is based on viscosity (diffusion), and $P(\bar{t})$ if the time scale is based on the pressure oscillations.
Filename: `pipeflow`.

Exercise 3.4: The linear cable equation

A key PDE in neuroscience is the cable equation, here given in its simplest linear form:

$$\tau\frac{\partial u}{\partial t} = \lambda^2\frac{\partial^2 u}{\partial x^2} - u\,. \tag{3.83}$$

The unknown u is the voltage (measured in volt) associated with an electric current along one-dimensional dendrites ("cables") in neural networks, while τ and λ are given parameters.

Scale (3.83) in three ways: 1) let all terms in the scaled equation have unit coefficients, 2) use the domain size L as spatial scale and base the time scale on diffusion, 3) use the domain size L as spatial scale and base the time scale on reaction, i.e., the $-u$ term.

Solution. Straightforward scaling, with scales u_c, t_c, and x_c, leads in the first step to

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{t_c\lambda^2}{\tau x_c^2}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} - \frac{t_c}{\tau}\bar{u}\,.$$

Assuming that all terms are equally important and of unit size in the scaled PDE, we get a uniquely determined length and space scale:

$$t_c = \tau, \quad x_c = \lambda\,.$$

The scaled cable equation is then

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{x}^2} - \bar{u}\,.$$

Let now the spatial scale be fixed as $x_c = L$. Basing t_c on diffusion means $t_c = \tau(L/\lambda)^2$, and the scaled PDE becomes

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{x}^2} - \beta\bar{u},$$

where

$$\beta = \left(\frac{L}{\lambda}\right)^2\,.$$

Basing t_c on the reaction scale, i.e., the balance of the time derivative and the reaction term, gives $t_c = \tau$ and the scaled PDE

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \beta^{-1}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} - \bar{u}\,.$$

In neuroscience applications of the cable equation to dendrites, it appears that λ is about 1 mm and of the same order of magnitude as the cable length, so β is around 1 in size. Then there are not big differences in these scalings, and the first one is to be preferred. The two others are more suitable when β is small or large, e.g., such that the term with β can be left out of the PDE. Filename: `cable_eq`.

Exercise 3.5: Heat conduction with discontinuous initial condition

Two pieces of metal at different temperature are brought in contact at $t = 0$. The following initial-boundary value problem governs the temperature evolution in the two pieces:

$$\frac{\partial u}{\partial t} = \alpha\nabla^2 u, \qquad \boldsymbol{x} \in \Omega,\ t \in (0, T], \tag{3.84}$$

$$u(\boldsymbol{x}, 0) = I(x), \qquad \boldsymbol{x} \in \Omega, \tag{3.85}$$

$$-\alpha\frac{\partial u}{\partial n} = h(u - u_S), \qquad x \in \partial\Omega,\ t \in (0, T]. \tag{3.86}$$

Here, $u(\boldsymbol{x},t)$ is the temperature, α the effective heat diffusion coefficient (assuming both pieces are homogeneous and of the same type of metal), and u_S is the surrounding temperature. The domain Ω consists of the two pieces Ω_1 and Ω_2: $\Omega = \Omega_1 \cup \Omega_2$. The initial condition can be specified as

$$I(x) = \begin{cases} U_1,\, \boldsymbol{x} \in \Omega_1, \\ U_2,\, \boldsymbol{x} \in \Omega_2, \end{cases}$$

where U_1 and U_2 are the constant initial temperatures in each piece.

Thinking of two identical pieces Ω_1 and Ω_2 with shapes as bricks, it is tempting to develop a one-dimensional model, especially if the pieces are somewhat slender. We then expect the main temperature variations to take place in the x direction, where the x axis is perpendicular to the contact surface between the pieces. A simplified PDE problem, neglecting variations in the y and z directions, takes the form

$$\frac{\partial v}{\partial t} = \alpha \frac{\partial^2 v}{\partial x^2} - \frac{hP}{A}(v(x,t) - u_S), \qquad x \in (0,L),\ t \in (0,T], \tag{3.87}$$

$$v(x,0) = I(x), \qquad x \in (0,L), \tag{3.88}$$

$$\alpha \frac{\partial v}{\partial x} = h(v(x,t) - u_S), \qquad x = 0,\ t \in (0,T], \tag{3.89}$$

$$-\alpha \frac{\partial v}{\partial x} = h(v(x,t) - u_S), \qquad x = L,\ t \in (0,T], \tag{3.90}$$

with

$$I(x) = \begin{cases} U_1,\, x \in [0,L/2), \\ U_2,\, x \in [L/2,L]\,. \end{cases}$$

The parameter P is the perimeter of the cross section and A is the area of the cross section. Scale this problem.

Solution. We expect the temperature to start from the discontinuous state with U_1 and U_2 and approach the surrounding temperature u_S in the cooling law as $t \to \infty$. One suitable scaling is then

$$\bar{v} = \frac{v - \min(U_1,U_2)}{u_S - \min(U_1,U_2)},$$

since this implies that u varies from 0 to 1. Without loss of generality we number the bricks such that $U_1 < U_2$, so

$$\bar{v} = \frac{v - U_1}{u_S - U_1}\,.$$

Furthermore,

$$\bar{x} = \frac{x}{L}, \quad \bar{t} = \frac{t}{t_c}\,.$$

Inserted in the governing PDE, we have

$$\frac{\partial \bar{v}}{\partial \bar{t}} = \frac{t_c\alpha}{L^2}\frac{\partial^2 \bar{v}}{\partial \bar{x}^2} - \frac{t_c hP}{A(u_S - U_1)}(U_1 + (u_S - U_1)\bar{v}(x,t) - u_S),$$

which simplifies to

$$\frac{\partial \bar{v}}{\partial \bar{t}} = \frac{t_c\alpha}{L^2}\frac{\partial^2 \bar{v}}{\partial \bar{x}^2} - \frac{t_c hP}{A}(\bar{v}(x,t) - 1)\,.$$

The natural time scale is that of diffusion: $t_c = L^2/\alpha$. This choice results in the scaled PDE

$$\frac{\partial \bar{v}}{\partial \bar{t}} = \frac{\partial^2 \bar{v}}{\partial \bar{x}^2} - \overline{\mathrm{Bi}}\,(\bar{v}(x,t) - 1),$$

where the dimensionless number

$$\overline{\mathrm{Bi}} = \frac{hL^2P}{\alpha A}$$

can be interpreted as a modified Biot number (if $A/P = L$, it would an ordinary Biot number, but here this is not an appropriate approximation: for bricks with square cross section of length a, $A/P = a/4$, and for a circular cross section, $A/P = R/2$). This modified Biot number governs the significance of lateral heat loss/gain to/from the environment.

The scaled initial condition becomes

$$\bar{v}(\bar{x},0) = 0 \text{ if } 0 \le x < 1/2 \text{ else } \beta,$$

where β is the dimensionless number

$$\beta = \frac{U_2 - U_1}{U_s - U_1}\,.$$

The scaled boundary conditions take the form

$$\frac{\partial}{\partial \bar{x}}\bar{v}(0,t) = \mathrm{Bi}\,(\bar{v}(0,t) - 1), \quad -\frac{\partial}{\partial \bar{x}}\bar{v}(L,t) = \mathrm{Bi}\,(\bar{v}(L,t) - 1),$$

where Bi is the standard Biot number for heat conduction from solids:

$$\mathrm{Bi} = \frac{Lh}{\alpha}\,.$$

Here is a simulation with $\mathrm{Bi} = 0.01$, $\overline{\mathrm{Bi}} = 0.2$, and $\beta = 1.5$ (using the `diffusion_two_metal_pieces` function in `session.py`).

Movie 4: `https://github.com/hplgit/scaling-book/raw/master/doc/pub/book/html/mov-scaling/metal_pieces/movie.mp4`

Another temperature scaling is also possible:

$$\bar{v} = \frac{v - U_s}{U_2 - U_s}\,.$$

The initial condition is then

$$\bar{v}(\bar{x},0) = \gamma \text{ if } 0 \leq x < 1/2 \text{ else } 1,$$

where γ is the dimensionless number

$$\gamma = \frac{U_1 - U_s}{U_2 - U_s}\,.$$

Note that $\gamma < 1$ if $U_1 < U_2$, and we expect $\bar{v} \in [0,1]$ ($\bar{v} \to 0$ as $\bar{t} \to \infty$). The PDE now becomes homogeneous,

$$\frac{\partial \bar{v}}{\partial \bar{t}} = \frac{\partial^2 \bar{v}}{\partial \bar{x}^2} - \overline{\mathrm{Bi}}\,\bar{v}(x,t),$$

and the boundary conditions take the form

$$\frac{\partial}{\partial \bar{x}}\bar{v}(0,t) = \mathrm{Bi}\,\bar{v}(0,t), \quad -\frac{\partial}{\partial \bar{x}}\bar{v}(L,t) = \mathrm{Bi}\,\bar{v}(L,t)\,.$$

Many will find the homogeneous PDE and boundary conditions of the latter scaling attractive, especially for analytical solution of the problem.
Filename: `metal_pieces`.

Remarks. We can derive (3.87)-(3.90) from (3.85)-(3.86). The idea is to integrate the governing PDE (3.87) in the two directions where we expect negligible variations, use the Gauss divergence theorem in these directions, and apply the cooling boundary condition. Let A be the cross section of the bricks. Integrating over A gives

$$\begin{aligned}
\int_A \frac{\partial u}{\partial t} dydz &= \int_A \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) dydz \\
&= \int_A \alpha \frac{\partial^2 u}{\partial x^2} dydz + \int_A \alpha \left(\frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) dydz \\
&= \int_A \alpha \frac{\partial^2 u}{\partial x^2} dydz + \alpha \int_{\partial A} \frac{\partial u}{\partial n} ds \\
&= \int_A \alpha \frac{\partial^2 u}{\partial x^2} dydz - h(v(x,t) - u_S)P\,.
\end{aligned}$$

The parameter P is the perimeter of the cross section A. The function $v(x,t)$ means $u(\boldsymbol{x},t)$ evaluated at the boundary ∂A. Assuming u to vary little across the cross section A, we can approximate the integrals by u evaluated at ∂A as v:

$$\int_A \frac{\partial u}{\partial t} dydz \approx A \frac{\partial}{\partial t} v(x,t), \quad \int_A \alpha \frac{\partial^2 u}{\partial x^2} dydz \approx A\alpha \frac{\partial^2 v}{\partial x^2},$$

where A now is the cross-section *area*. The result is the 1D initial-boundary value problem (3.87)-(3.90).

Problem 3.6: Scaling a welding problem

Welding equipment makes a very localized heat source that moves in time. We shall investigate the heating due to welding and choose, for maximum simplicity, a one-dimensional heat equation with a fixed temperature at the ends (a 2D or 3D model with cooling conditions at the boundaries would be of greater physical significance, but now the scaling is in focus). The effect of melting is not included in the heat equation. Our goal is to investigate two alternative scalings through numerical experimentation.

The governing PDE problem reads

$$\begin{aligned}
\varrho c \frac{\partial u}{\partial t} &= k \frac{\partial^2 u}{\partial x^2} + f, && x \in (0,L),\ t \in (0,T),\\
u(x,0) &= U_s, && x \in [0,L],\\
u(0,t) = u(L,t) &= U_s, && t \in (0,T].
\end{aligned}$$

Here, u is the temperature, ϱ the density of the material, c a heat capacity, k the heat conduction coefficient, f is the heat source from the welding equipment, and U_s is the initial constant (room) temperature in the material.

A possible model for the heat source is a moving Gaussian function:

$$f = A \exp\left(-\frac{1}{2}\left(\frac{x - vt}{\sigma}\right)^2\right),$$

where A is the strength, σ is a parameter governing how peak-shaped (or localized in space) the heat source is, and v is the velocity (in positive x direction) of the source.

a) Let x_c, t_c, u_c, and f_c be scales, i.e., characteristic sizes, of x, t, u, and f, respectively. The natural choice of x_c and f_c is L and A, since these make the scaled x and f in the interval $[0,1]$. If each of the three terms in the PDE are equally important, we can find t_c and u_c by demanding that the coefficients in the scaled PDE are all equal to unity. Perform this scaling. Use scaled quantities in the arguments for the exponential function in f too and show that

$$\bar{f} = \exp\left(-\frac{1}{2}\beta^2(\bar{x}-\gamma\bar{t})^2\right),$$

where β and γ are dimensionless numbers. Give an interpretation of β and γ.

Solution. We introduce

$$\bar{x} = \frac{x}{L}, \quad \bar{t} = \frac{t}{t_c}, \quad \bar{u} = \frac{u-U_s}{u_c}, \quad \bar{f} = \frac{f}{A}\,.$$

Inserted in the PDE and dividing by $\varrho c u_c/t_c$ such that the coefficient in front of $\partial\bar{u}/\partial\bar{t}$ becomes unity, and thereby all terms become dimensionless, we get

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{k t_c}{\varrho c L^2}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \frac{A t_c}{\varrho c u_c}\bar{f}\,.$$

Demanding that all three terms are equally important, it follows that

$$\frac{k t_c}{\varrho c L^2} = 1, \quad \frac{A t_c}{\varrho c u_c} = 1\,.$$

These constraints imply the diffusion time scale

$$t_c = \frac{\varrho c L^2}{k},$$

and a scale for u_c,

$$u_c = \frac{A L^2}{k}\,.$$

The scaled PDE reads

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \bar{f}\,.$$

Scaling f results in

$$\begin{aligned}
\bar{f} &= \exp\left(-\frac{1}{2}\left(\frac{x-vt}{\sigma}\right)^2\right)\\
&= \exp\left(-\frac{1}{2}\frac{L^2}{\sigma^2}\left(\bar{x}-\frac{v t_c}{L}\bar{t}\right)^2\right)\\
&= \exp\left(-\frac{1}{2}\beta^2\left(\bar{x}-\gamma\bar{t}\right)^2\right),
\end{aligned}$$

where β and γ are dimensionless numbers:

$$\beta = \frac{L}{\sigma}, \quad \gamma = \frac{v t_c}{L} = \frac{v \varrho c L}{k}\,.$$

The σ parameter measures the width of the Gaussian peak, so β is the ratio of the domain and the width of the heat source (large β implies a very peak-formed heat source). The γ parameter arises from $t_c/(L/v)$, which is the ratio of the diffusion time scale and the time it takes for the heat source to travel through the domain. Equivalently, we can multiply by t_c/t_c to get $\gamma = v/(t_c L)$ as the ratio between the velocity of the heat source and the diffusion velocity.

b) Argue that at least for large γ we should base the time scale on the movement of the heat source. Using L as length scale, show that this gives rise to the scaled PDE

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \gamma^{-1}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \bar{f},$$

and

$$\bar{f} = \exp\left(-\frac{1}{2}\beta^2(\bar{x} - \bar{t})^2\right).$$

Discuss when the scalings in a) and b) are appropriate.

Solution. We perform the scaling as in a), but this time we determine t_c such that the heat source moves with unit velocity. This means that

$$\frac{vt_c}{L} = 1 \quad \Rightarrow \quad t_c = \frac{L}{v}.$$

Scaling of the PDE gives, as before,

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{kt_c}{\varrho c L^2}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \frac{At_c}{\varrho c u_c}\bar{f}.$$

Inserting the expression for t_c, we have

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{kL}{\varrho c L^2 v}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \frac{AL}{v\varrho c u_c}\bar{f}.$$

We recognize the first coefficient as γ^{-1}, while u_c can be determined from demanding the second coefficient to be unity:

$$u_c = \frac{AL}{v\varrho c}.$$

The scaled PDE is therefore

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \gamma^{-1}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \bar{f}.$$

If the heat source moves very fast, there is little time for the diffusion to transport the heat away from the source, and the heat conduction term becomes insignificant. This is reflected in the coefficient γ^{-1}, which is small when γ, the ratio of the heat source velocity and the diffusion velocity, is large.

The scaling in a) is therefore appropriate if diffusion is a significant process, i.e., the welding equipment moves at a slow speed so heat can efficiently spread out by diffusion. For large γ, the scaling in b) is appropriate, and $t = 1$ corresponds to having the heat source traveled through the domain (with the scaling in a), the heat source will leave the domain in short time).

c) For fast movement of the welding equipment, i.e., when heat transfer is less important than the local heating by the equipment, the typical length scale of the local heating is the size of the source, reflected by the σ parameter. Modify the scaling in b) when σ is chosen as length scale.

Solution. With σ as length scale, the scaled PDE has the initial form

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{kt_c}{\varrho c\sigma^2}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \frac{At_c}{\varrho cu_c}\bar{f}\,.$$

The scaling of f becomes

$$\begin{aligned}\bar{f} &= \exp\left(-\frac{1}{2}\left(\frac{x-vt}{\sigma}\right)^2\right)\\ &= \exp\left(-\frac{1}{2}\left(\bar{x}-\frac{vt_c}{\sigma}\bar{t}\right)^2\right)\\ &= \exp\left(-\frac{1}{2}\left(\bar{x}-\bar{t}\right)^2\right),\end{aligned}$$

where we have chosen t_c as σ/v. Inserting t_c in the PDE leads to a coefficient

$$\frac{k}{v\varrho c\sigma} = \gamma^{-1}\beta\,.$$

As before, we chose u_c to make the other coefficient equal unity, modulo a scaling factor. The $\gamma^{-1}\beta$ parameter is the ratio of the moving heat source time scale v/σ and the diffusion time scale $\sigma^2/(k/\varrho c)$.

The equations are as in), except that $\beta = 1$ and ϵ replaces γ. Moreover, the length of the domain is now L/σ, i.e., β enters the problem in the domain size.

d) A fourth kind of possible scaling is to say that for small γ, the problem is quasi-stationary and the heat transfer balances the heat source. Determine u_c from this assumption. Use L as length scale and a time scale as in b), i.e., based on the movement of the welding equipment.

Solution. We insert the dimensionless quantities in the PDE, but this time we make the factor in the source term unity:

$$\frac{\varrho cu_c}{At_c}\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{ku_c}{AL^2}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} + \bar{f}\,.$$

Assuming the two terms on the right-hand side balance, we must have $ku_c/(AL^2) = 1$ and hence $u_c = L^2A/k$. This gives the coefficient $\varrho cL^2/(kt_c)$ on the left-hand side. From b) we have that a time scale based the movement of the heat source: $t_c = L/v$. Now the scaled PDE becomes

$$\gamma\frac{\partial\bar{u}}{\partial\bar{t}} = \frac{\partial^2\bar{u}}{\partial\bar{x}^2} + \bar{f}\,.$$

The scaling of f becomes identical to the one in b).

e) One aim with scaling is to get a solution that lies in the interval $[-1,1]$. This is not always the case when u_c is based on a scale involving a source term, as we do in a)-c). However, from the scaled PDE we realize that if we replace $\bar{f}$ with $\delta\bar{f}$, where δ is a dimensionless factor, this corresponds to replacing u_c by u_c/δ. So, if we observe that $\bar{u} \sim 1/\delta$ in simulations, we can just replace $\bar{f}$ by $\delta\bar{f}$ in the scaled PDE.

Use this trick and implement the four scaled models in a)-d). Reuse some software for the 1D diffusion equation. Make a function `run(gamma, beta=10, delta=40, scaling=1, animate=False)` that runs an implementation of the unscaled model with the given γ, β, and δ parameters as well as an indicator `scaling` that is `'a'`, `'b'`, and so forth. The last argument can be used to turn screen animations on or off.

Perform experiments to find the proper value of δ for each γ and for each scaling.

Equip the `run` function with visualization, both animation of $\bar{u}$ and $\bar{f}$, and plots with $\bar{u}$ and $\bar{f}$ for $t = 0.2$ and $t = 0.5$.

Hint. Since the amplitudes of $\bar{u}$ and $\bar{f}$ differs by a factor δ, it is attractive to plot $\bar{f}/\delta$ together with $\bar{u}$.

Solution. Here is a possible general `solver` function for solving the 1D diffusion equation

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(\alpha(x)\frac{\partial u}{\partial x}\right) + f,$$

by the θ-rule in time and centered finite differences in space. The θ-rule in time is actually just a notational convenience that gives Forward Euler explicit time stepping for $\theta = 0$, Backward Euler implicit time stepping for $\theta = 0$, and a centered implicit Crank-Nicolson scheme for $\theta = \frac{1}{2}$. The boundary conditions is of Dirichlet type: $u(0,t) = U_L(t)$ and $u(L,t) = U_R(t)$.

```
import numpy as np
import scipy.sparse
import scipy.sparse.linalg
import time, sys


def solver(I, a, f, L, Nx, D, T, theta=0.5, u_L=1, u_R=0,
           user_action=None):
```

```
    """
    The a variable is an array of length Nx+1 holding the values of
    a(x) at the mesh points.

    Method: (implicit) theta-rule in time.

    Nx is the total number of mesh cells; mesh points are numbered
    from 0 to Nx.
    D = dt/dx**2 and implicitly specifies the time step.
    T is the stop time for the simulation.
    I is a function of x.

    user_action is a function of (u, x, t, n) where the calling code
    can add visualization, error computations, data analysis,
    store solutions, etc.
    """
    import time
    t0 = time.clock()

    x = np.linspace(0, L, Nx+1)   # mesh points in space
    dx = x[1] - x[0]
    dt = D*dx**2
    #print 'dt=%g' % dt
    Nt = int(round(T/float(dt)))
    t = np.linspace(0, T, Nt+1)   # mesh points in time

    if isinstance(a, (float,int)):
        a = np.zeros(Nx+1) + a
    if isinstance(u_L, (float,int)):
        u_L_ = float(u_L)  # must take copy of u_L number
        u_L = lambda t: u_L_
    if isinstance(u_R, (float,int)):
        u_R_ = float(u_R)  # must take copy of u_R number
        u_R = lambda t: u_R_

    u   = np.zeros(Nx+1)   # solution array at t[n+1]
    u_1 = np.zeros(Nx+1)   # solution at t[n]

    """
    Basic formula in the scheme:

    0.5*(a[i+1] + a[i])*(u[i+1] - u[i]) -
    0.5*(a[i] + a[i-1])*(u[i] - u[i-1])

    0.5*(a[i+1] + a[i])*u[i+1]
    0.5*(a[i] + a[i-1])*u[i-1]
    -0.5*(a[i+1] + 2*a[i] + a[i-1])*u[i]
    """
    Dl = 0.5*D*theta
    Dr = 0.5*D*(1-theta)

    # Representation of sparse matrix and right-hand side
    diagonal = np.zeros(Nx+1)
    lower    = np.zeros(Nx)
```

```
    upper    = np.zeros(Nx)
    b        = np.zeros(Nx+1)

    # Precompute sparse matrix (scipy format)
    diagonal[1:-1] = 1 + Dl*(a[2:] + 2*a[1:-1] + a[:-2])
    lower[:-1] = -Dl*(a[1:-1] + a[:-2])
    upper[1:]  = -Dl*(a[2:] + a[1:-1])
    # Insert boundary conditions
    diagonal[0] = 1
    upper[0] = 0
    diagonal[Nx] = 1
    lower[-1] = 0

    A = scipy.sparse.diags(
        diagonals=[diagonal, lower, upper],
        offsets=[0, -1, 1],
        shape=(Nx+1, Nx+1),
        format='csr')
    #print A.todense()

    # Set initial condition
    for i in range(0,Nx+1):
        u_1[i] = I(x[i])

    if user_action is not None:
        user_action(u_1, x, t, 0)

    # Time loop
    for n in range(0, Nt):
        b[1:-1] = u_1[1:-1] + Dr*(
            (a[2:] + a[1:-1])*(u_1[2:] - u_1[1:-1]) -
            (a[1:-1] + a[0:-2])*(u_1[1:-1] - u_1[:-2])) + \
            dt*theta*f(x[1:-1], t[n+1]) + \
            dt*(1-theta)*f(x[1:-1], t[n])
        # Boundary conditions
        b[0]  = u_L(t[n+1])
        b[-1] = u_R(t[n+1])
        # Solve
        u[:] = scipy.sparse.linalg.spsolve(A, b)

        if user_action is not None:
            user_action(u, x, t, n+1)

        # Switch variables before next step
        u_1, u = u, u_1

    t1 = time.clock()
    return t1-t0
```

And here is our `run` function tailored to the problem:

```
def run(gamma, beta=10, delta=40, scaling=1, animate=False):
    """Run the scaled model for welding."""
    gamma = float(gamma)  # avoid integer division
```

```
    if scaling == 'a':
        v = gamma
        a = 1
        L = 1.0
        b = 0.5*beta**2
    elif scaling == 'b':
        v = 1
        a = 1.0/gamma
        L = 1.0
        b = 0.5*beta**2
    elif scaling == 'c':
        v = 1
        a = beta/gamma
        L = beta
        b = 0.5
    elif scaling == 'd':
        # PDE: u_t = gamma**(-1)u_xx + gamma**(-1)*delta*f
        v = 1
        a = 1.0/gamma
        L = 1.0
        b = 0.5*beta**2
        delta *= 1.0/gamma

    ymin = 0
    # Need global ymax to be able change ymax in closure process_u
    global ymax
    ymax = 1.2

    I = lambda x: 0
    f = lambda x, t: delta*np.exp(-b*(x - v*t)**2)

    import time
    import scitools.std as plt
    plot_arrays = []
    if scaling == 'c':
        plot_times = [0.2*beta, 0.5*beta]
    else:
        plot_times = [0.2, 0.5]

    def process_u(u, x, t, n):
        """
        Animate u, and store arrays in plot_arrays if
        t coincides with chosen times for plotting (plot_times).
        """
        global ymax
        if animate:
            plt.plot(x, u, 'r-',
                     x, f(x, t[n])/delta, 'b-',
                     axis=[0, L, ymin, ymax], title='t=%f' % t[n],
                     xlabel='x', ylabel='u and f/%g' % delta)
        if t[n] == 0:
            time.sleep(1)
            plot_arrays.append(x)
        dt = t[1] - t[0]
```

```
        tol = dt/10.0
        if abs(t[n] - plot_times[0]) < tol or \
           abs(t[n] - plot_times[1]) < tol:
            plot_arrays.append((u.copy(), f(x, t[n])/delta))
            if u.max() > ymax:
                ymax = u.max()

    Nx = 100
    D = 10
    if scaling == 'c':
        T = 0.5*beta
    else:
        T = 0.5
    u_L = u_R = 0
    theta = 1.0
    cpu = solver(
        I, a, f, L, Nx, D, T, theta, u_L, u_R, user_action=process_u)
    x = plot_arrays[0]
    plt.figure()
    for u, f in plot_arrays[1:]:
        plt.plot(x, u, 'r-', x, f, 'b--', axis=[x[0], x[-1], 0, ymax],
                 xlabel='$x$', ylabel=r'$u, \ f/%g$' % delta)
        plt.hold('on')
    plt.legend(['$u,\\ t=%g$' % plot_times[0],
                '$f/%g,\\ t=%g$' % (delta, plot_times[0]),
                '$u,\\ t=%g$' % plot_times[1],
                '$f/%g,\\ t=%g$' % (delta, plot_times[1])])
    filename = 'tmp1_gamma%g_%s' % (gamma, scaling)
    plt.title(r'$\beta = %g,\ \gamma = %g,\ $' % (beta, gamma)
              + 'scaling=%s' % scaling)
    plt.savefig(filename + '.pdf');  plt.savefig(filename + '.png')
    return cpu
```

Note that we have dropped the bar notation in the plots. It is common to drop the bars as soon as the scaled problem is established.

f) Use the software in e) to investigate $\gamma = 0.2, 1, 5, 40$ for the four scalings. Discuss the results.

Solution. For these investigations, we compare the three scalings for each of the different γ values. An appropriate function for automating the tasks is

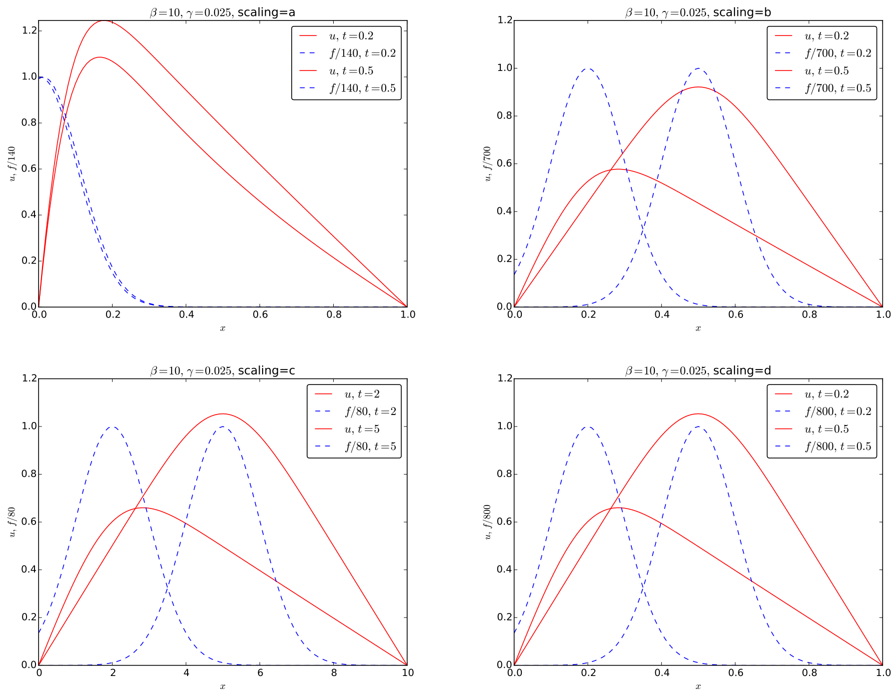
```
def investigate():
    """Do scienfic experiments with the run function above."""
    # Clean up old files
    import glob, os
    for filename in glob.glob('tmp1_gamma*') + \
            glob.glob('welding_gamma*'):
        os.remove(filename)

    scaling_values = 'abcd'
    gamma_values = 1, 40, 5, 0.2, 0.025
    delta_values = {}  # delta_values[scaling][gamma]
    delta_values['a'] = {0.025: 140, 0.2: 60,  1: 20, 5: 40,  40: 800}
```

```
    delta_values['b'] = {0.025: 700, 0.2: 100, 1: 20, 5: 8,   40: 5}
    delta_values['c'] = {0.025: 80,  0.2: 10,  1: 2,  5: 0.8, 40: 0.5}
    delta_values['d'] = {0.025: 20,  0.2: 20,  1: 20, 5: 40,  40: 200}
    for gamma in gamma_values:
        for scaling in scaling_values:
            run(gamma=gamma, beta=10,
                delta=delta_values[scaling][gamma],
                scaling=scaling)

    # Combine images
    for gamma in gamma_values:
        for ext in 'pdf', 'png':
            cmd = 'doconce combine_images -2'
            for s in scaling_values:
                cmd += ' tmp1_gamma%(gamma)g_%(s)s.%(ext)s ' % vars()
            cmd += ' welding_gamma%(gamma)g.%(ext)s' % vars()
            os.system(cmd)
            # pdflatex doesn't like a dot (as in 0.2) in filenames...
            if '.' in str(gamma):
                os.rename(
                'welding_gamma%(gamma)g.%(ext)s' % vars(),
                ('welding_gamma%(gamma)g' % vars()).replace('.', '_')
                + '.' + ext)
```

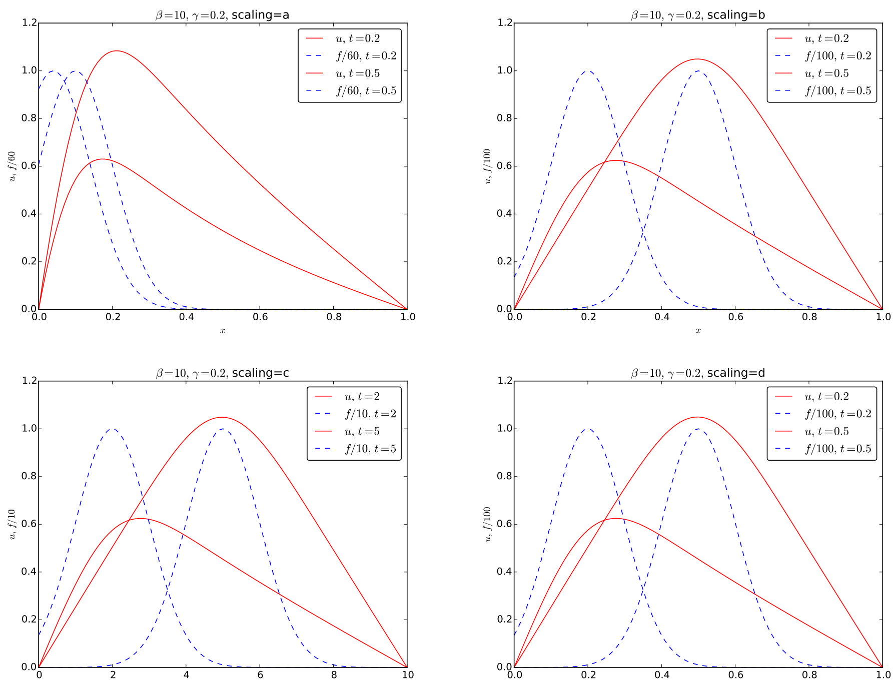
Note that for each γ value and each scaling, we have found a δ value such that the maximum u value is around unity in size. We did this first by trial and error, and thereafter filled out the `delta_values` dictionary.

The scalings in b)-d) are illustrated at the same *physical* times. The scaling in a) is plotted at the same non-dimensional time as used in the other scalings, but observe that this is a different physical time than used for the b)-d) scalings.

We get the following plots, with the a) and c) scalings to the left and the b) and d) scalings to the right, starting with $\gamma = 0.025$ and increasing its value to $\gamma = 40$:

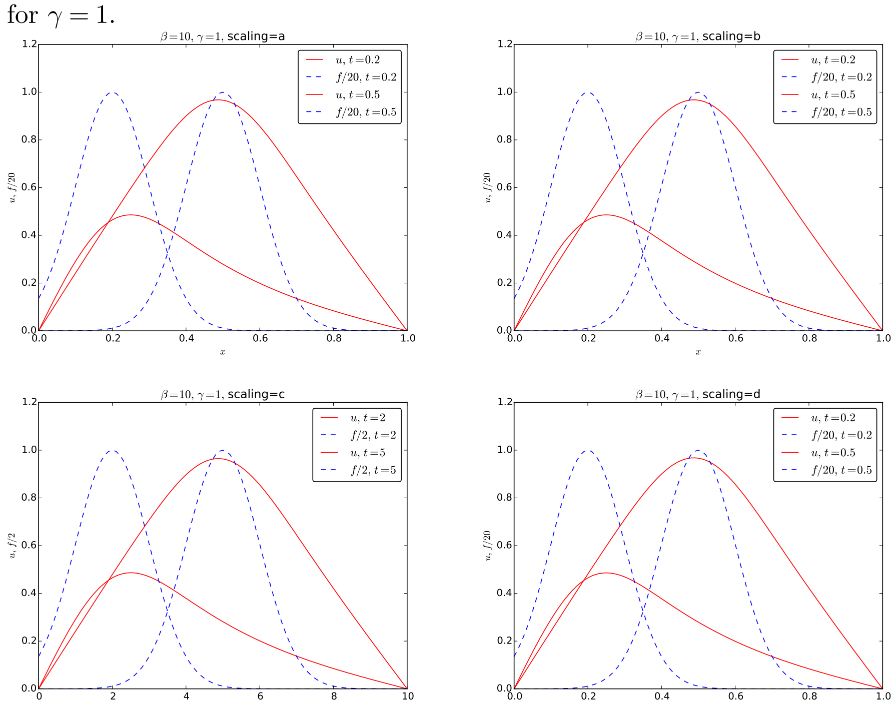

$\beta=10$, $\gamma=0.025$, scaling=a
u, $t=0.2$
$f/140$, $t=0.2$
u, $t=0.5$
$f/140$, $t=0.5$
u, $f/140$
x
$\beta=10$, $\gamma=0.025$, scaling=b
u, $t=0.2$
$f/700$, $t=0.2$
u, $t=0.5$
$f/700$, $t=0.5$
u, $f/700$
x
$\beta=10$, $\gamma=0.025$, scaling=c
u, $t=2$
$f/80$, $t=2$
u, $t=5$
$f/80$, $t=5$
u, $f/80$
x
$\beta=10$, $\gamma=0.025$, scaling=d
u, $t=0.2$
$f/800$, $t=0.2$
u, $t=0.5$
$f/800$, $t=0.5$
u, $f/800$
x

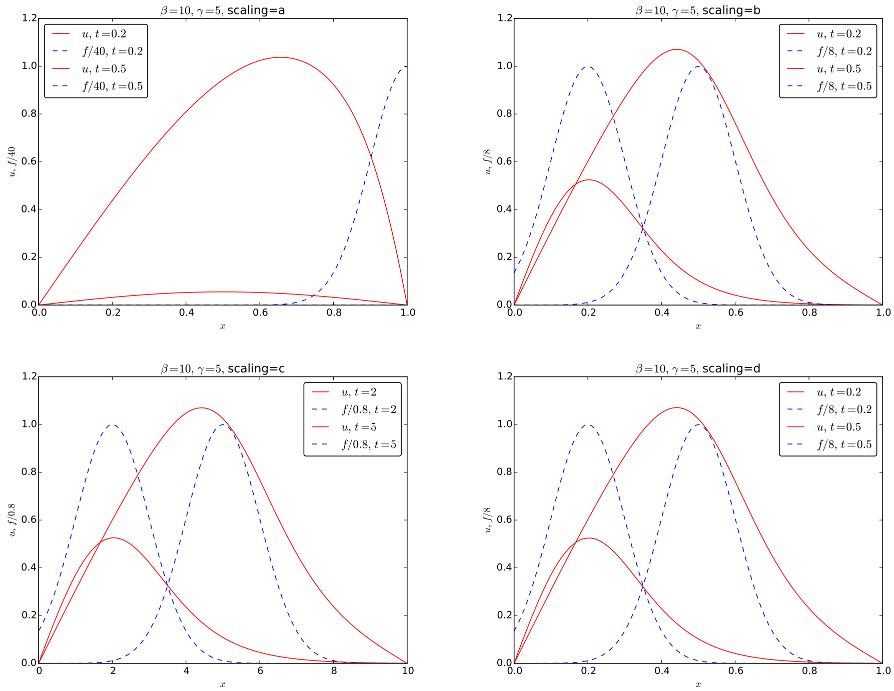
$\gamma = 0.025$.


$\beta=10$, $\gamma=0.2$, scaling=a
u, $t=0.2$
$f/60$, $t=0.2$
u, $t=0.5$
$f/60$, $t=0.5$
u, $f/60$
x
$\beta=10$, $\gamma=0.2$, scaling=b
u, $t=0.2$
$f/100$, $t=0.2$
u, $t=0.5$
$f/100$, $t=0.5$
u, $f/100$
x
$\beta=10$, $\gamma=0.2$, scaling=c
u, $t=2$
$f/10$, $t=2$
u, $t=5$
$f/10$, $t=5$
u, $f/10$
x
$\beta=10$, $\gamma=0.2$, scaling=d
u, $t=0.2$
$f/100$, $t=0.2$
u, $t=0.5$
$f/100$, $t=0.5$
u, $f/100$
x

$\gamma = 0.02$**.** For small γ, 0.025 and 0.2, the scales b)-d) are equally applicable, while the diffusion time scale, which is much longer,

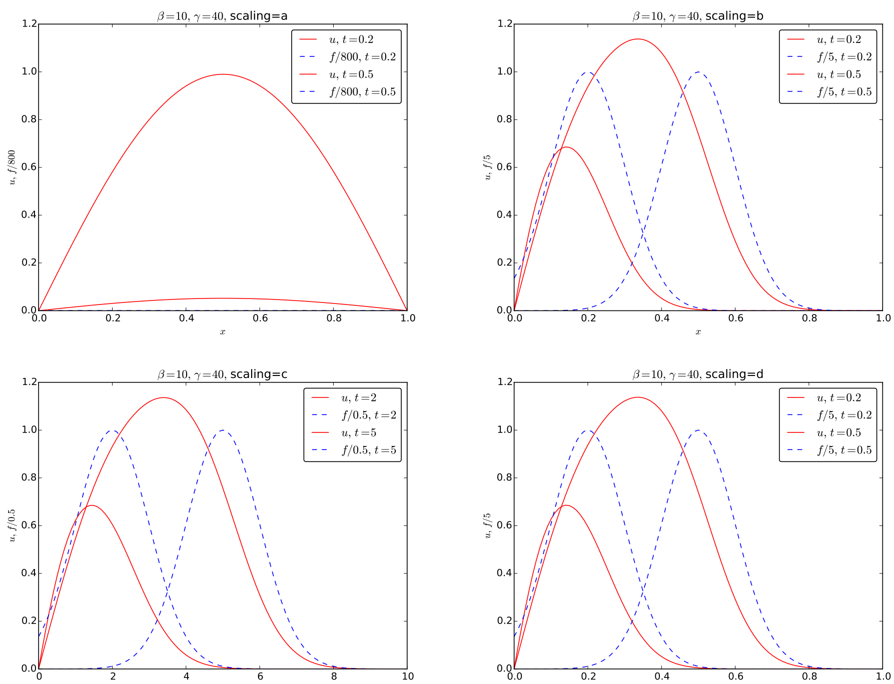
$\gamma = 1$**.** Not surprisingly, all the scalings are equal and provide identical results for $\gamma = 1$.





$\beta = 10$, $\gamma = 5$, scaling=a
u, $t = 0.2$
$f/40$, $t = 0.2$
u, $t = 0.5$
$f/40$, $t = 0.5$
u, $f/40$
x
$\beta = 10$, $\gamma = 5$, scaling=b
u, $t = 0.2$
$f/8$, $t = 0.2$
u, $t = 0.5$
$f/8$, $t = 0.5$
u, $f/8$
x
$\beta = 10$, $\gamma = 5$, scaling=c
u, $t = 2$
$f/0.8$, $t = 2$
u, $t = 5$
$f/0.8$, $t = 5$
u, $f/0.8$
x
$\beta = 10$, $\gamma = 5$, scaling=d
u, $t = 0.2$
$f/8$, $t = 0.2$
u, $t = 0.5$
$f/8$, $t = 0.5$
u, $f/8$
x

$\gamma = 5$.



$\beta = 10$, $\gamma = 40$, scaling=a
u, $t = 0.2$
$f/800$, $t = 0.2$
u, $t = 0.5$
$f/800$, $t = 0.5$
u, $f/800$
x
$\beta = 10$, $\gamma = 40$, scaling=b
u, $t = 0.2$
$f/5$, $t = 0.2$
u, $t = 0.5$
$f/5$, $t = 0.5$
u, $f/5$
x
$\beta = 10$, $\gamma = 40$, scaling=c
u, $t = 2$
$f/0.5$, $t = 2$
u, $t = 5$
$f/0.5$, $t = 5$
u, $f/0.5$
x
$\beta = 10$, $\gamma = 40$, scaling=d
u, $t = 0.2$
$f/5$, $t = 0.2$
u, $t = 0.5$
$f/5$, $t = 0.5$
u, $f/5$
x

$\gamma = 40$. From these plots we see that it does not matter which of the scalings in b)-d) we choose, as long as we compensate with the δ parameter to bring u to the size of unity. The scaling a) is suitable for $\gamma \leq 1$, it seems, but graphs at earlier times for $\gamma = 5, 40$ should be investigated before drawing a conclusion.

Looking at the necessary values of δ to bring u around the size of unity, and noting that the factor used to scale f is δ/γ in scaling d), we see that for small γ, the scaling in d) requires an adjustment $\delta = 20$, while the scaling in b) requires $\delta = 700$. For the large $\gamma = 40$, the situation is opposite: the scaling in d) requires $\delta = 200$, while the one in b) only needs $\delta = 5$. The scaling in c) is very similar to the scaling in b), apart from the factor L/σ (times, δ, etc. are all differ by the factor L/σ).

We therefore conclude that the scaling in d) is best for small γ and the scaling b) is best for large γ, but with the δ trick used here, it really does not matter for practical calculations which scaling we use. Nevertheless, the insight given by the scalings should not be forgotten: we see in d) that for large γ the time-derivative term can be neglected and in b) that the conduction term can be neglected.
Filename: `welding`.

Chapter 4
Advanced partial differential equation models

This final chapter addresses more complicated PDE models, including linear elasticity, viscous flow, heat transfer, porous media flow, gas dynamics, and electrophysiology. A range of classical dimensionless numbers are discussed in terms of the scaling.

4.1 The equations of linear elasticity

To the best of the authors' knowledge, it seems that mathematical models in elasticity and structural analysis are almost never non-dimensionalized. This is probably due to tradition, but the following sections will demonstrate the usefulness of scaling also in this scientific field.

We start out with the general, time-dependent elasticity PDE with variable material properties. Analysis based on scaling is used to determine under what circumstances the acceleration term can be neglected and we end up with the widely used stationary elasticity PDE. Scaling of different types of boundary conditions is also treated. At the end, we scale the equations of coupled thermo-elasticity. All the models make the assumption of small displacement gradients and Hooke's generalized constitutive law such that linear elasticity theory applies.

4.1.1 The general time-dependent elasticity problem

The following vector PDE governs deformation and stress in purely elastic materials, under the assumption of small displacement gradients:

$$\varrho\frac{\partial^2 \boldsymbol{u}}{\partial t^2} = \nabla((\lambda+\mu)\nabla\cdot\boldsymbol{u}) + \nabla\cdot(\mu\nabla\boldsymbol{u}) + \varrho\boldsymbol{f}\,. \tag{4.1}$$

Here, $\boldsymbol{u}$ is the displacement vector, ϱ is the density of the material, λ and μ are the Lame elasticity parameters, and $\boldsymbol{f}$ is a body force (gravity, centrifugal force, or similar).

We introduce dimensionless variables:

$$\bar{\boldsymbol{u}} = u_c^{-1}\boldsymbol{u}, \quad \bar{x} = \frac{x}{L} \quad \bar{y} = \frac{y}{L} \quad \bar{z} = \frac{z}{L}, \quad \bar{t} = \frac{f}{t_c}.$$

Also the elasticity parameters and the density can be scaled, if they are not constants,

$$\bar{\lambda} = \frac{\lambda}{\lambda_c}, \quad \bar{\mu} = \frac{\mu}{\mu_c}, \quad \bar{\varrho} = \frac{\varrho}{\varrho_c},$$

where the characteristic quantities are typically spatial maximum values of the functions:

$$\lambda_c = \max_{x,y,z}\lambda, \quad \mu_c = \max_{x,y,z}\mu, \quad \varrho_c = \max_{x,y,z}\varrho.$$

Finally, we scale $\boldsymbol{f}$ too (if not constant):

$$\bar{\boldsymbol{f}} = f_c^{-1}\boldsymbol{f}, \quad f_c = \max_{x,y,z,t}||\boldsymbol{f}||.$$

Inserting the dimensionless quantities in the governing vector PDE results in

$$\frac{\varrho_c u_c}{t_c^2}\frac{\partial^2\bar{\boldsymbol{u}}}{\partial\bar{t}^2} = L^{-2}u_c\bar{\nabla}((\lambda_c\bar{\lambda} + \mu_c\bar{\mu})\bar{\nabla}\cdot\bar{\boldsymbol{u}}) + L^{-2}u_c\mu_c\bar{\nabla}\cdot(\bar{\mu}\bar{\nabla}\bar{\boldsymbol{u}}) + \varrho_c f_c\bar{\varrho}\bar{\boldsymbol{f}}.$$

Making the terms non-dimensional gives the equation

$$\bar{\varrho}\frac{\partial^2\bar{\boldsymbol{u}}}{\partial\bar{t}^2} = \frac{t_c^2\lambda_c}{L^2\varrho_c}\bar{\nabla}(\bar{\lambda}\bar{\nabla}\cdot\bar{u}) + \frac{t_c^2\mu_c}{L^2\varrho_c}\bar{\nabla}(\bar{\mu}\bar{\nabla}\cdot\bar{\boldsymbol{u}}) + \frac{t_c^2\mu_c}{L^2\varrho_c}\bar{\nabla}\cdot(\bar{\mu}\bar{\nabla}\bar{\boldsymbol{u}}) + \frac{t_c^2 f_c}{u_c}\bar{\varrho}\bar{\boldsymbol{f}}. \quad (4.2)$$

We may choose t_c to make the coefficient in front of any of the spatial derivative terms equal unity. Here we choose the μ term, which implies

$$t_c = L\sqrt{\frac{\varrho_c}{\mu_c}}.$$

The scale for $\boldsymbol{u}$ can be chosen from an initial displacement or by making the coefficient in front of the $\bar{\boldsymbol{f}}$ term unity. The latter means

$$u_c = \mu_c^{-1}\varrho_c f_c L^2.$$

As discussed later, in Section 4.1.4, this might not be the desired u_c in applications.

The resulting dimensionless PDE becomes

$$\bar{\varrho}\frac{\partial^2 \bar{\boldsymbol{u}}}{\partial \bar{t}^2} = \bar{\nabla}((\beta\bar{\lambda} + \bar{\mu})\bar{\nabla}\cdot\bar{u}) + \bar{\nabla}\cdot(\bar{\mu}\bar{\nabla}\bar{\boldsymbol{u}}) + \bar{\varrho}\bar{\boldsymbol{f}}\,. \tag{4.3}$$

The only dimensionless parameter is

$$\beta = \frac{\lambda_c}{\mu_c}\,.$$

If the source term is absent, we must use the initial condition or a known boundary displacement to determine u_c.

Software. Given software for (4.1), we can simulate the dimensionless problem by setting $\varrho = \bar{\varrho}$, $\lambda = \beta\bar{\lambda}$, and $\mu = \bar{\mu}$.

4.1.2 Dimensionless stress tensor

The stress tensor $\boldsymbol{\sigma}$ is a key quantity in elasticity and is given by

$$\boldsymbol{\sigma} = \lambda\nabla\cdot\boldsymbol{u}\boldsymbol{I} + \mu(\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T)\,.$$

This $\boldsymbol{\sigma}$ can be computed as soon as the PDE problem for $\boldsymbol{u}$ has been solved. Inserting dimensionless variables on the right-hand side of the above relation gives

$$\begin{aligned}\boldsymbol{\sigma} &= \lambda_c u_c L^{-2}\bar{\lambda}\bar{\nabla}\cdot\bar{\boldsymbol{u}} + \mu_c u_c L^{-1}\bar{\mu}(\bar{\nabla}\bar{\boldsymbol{u}} + (\bar{\nabla}\bar{\boldsymbol{u}})^T)\\ &= \mu_c u_c L^{-1}\left(\beta\bar{\lambda}\bar{\nabla}\cdot\bar{\boldsymbol{u}} + \bar{\mu}(\bar{\nabla}\bar{\boldsymbol{u}} + (\bar{\nabla}\bar{\boldsymbol{u}})^T)\right)\,.\end{aligned}$$

The coefficient on the right-hand side, $\mu_c u_c L^{-1}$, has dimension of stress, since (according to the second table in Section 1.1.2) $[\mathrm{MT}^{-2}\mathrm{L}^{-1})(\mathrm{L})(\mathrm{L}^{-1})] = [\mathrm{MT}^{-2}\mathrm{L}^{-1}]$, which is the dimension of stress. The quantity $\mu_c u_c L^{-1}$ is therefore the natural scale of the stress tensor:

$$\bar{\boldsymbol{\sigma}} = \frac{\boldsymbol{\sigma}}{\sigma_c}, \quad \sigma_c = \mu_c u_c L^{-1},$$

and we have the dimensionless stress-displacement relation

$$\bar{\boldsymbol{\sigma}} = \beta\bar{\lambda}\bar{\nabla}\cdot\bar{\boldsymbol{u}} + \bar{\mu}(\bar{\nabla}\bar{\boldsymbol{u}} + (\bar{\nabla}\bar{\boldsymbol{u}})^T)\,. \tag{4.4}$$

4.1.3 When can the acceleration term be neglected?

A lot of applications of the elasticity equation involve static or quasi-static deformations where the acceleration term $\varrho\boldsymbol{u}_{tt}$ is neglected. Now we shall see under which conditions the quasi-static approximation holds.

The further discussion will need to look into the time scales of elastic waves, because it turns out that the chosen t_c above is closely linked to the propagation speed of elastic waves in a homogeneous body without body forces. A relevant model for such waves has constant ϱ, λ, and μ, and no force term:

$$\varrho\frac{\partial^2\boldsymbol{u}}{\partial t^2} = (\lambda+\mu)\nabla\nabla\cdot\boldsymbol{u} + \mu\nabla^2\boldsymbol{u}. \tag{4.5}$$

S waves. Let us take the curl of this PDE and notice that the curl of a gradient vanishes. The result is

$$\frac{\partial^2}{\partial t^2}\nabla\times\boldsymbol{u} = c_S^2\nabla^2\nabla\times\boldsymbol{u},$$

i.e., a wave equation for $\nabla\times\boldsymbol{u}$. The wave velocity is

$$c_S = \sqrt{\frac{\mu}{\varrho}}.$$

The corresponding waves are called S waves. The curl of a displacement field is closely related to rotation of continuum elements. S waves are therefore rotation waves, also sometimes referred to as shear waves.

The divergence of a displacement field can be interpreted as the volume change of continuum elements. Suppose this volume change vanishes, $\nabla\cdot\boldsymbol{u} = 0$, which means that the material is incompressible. The elasticity equation then simplifies to

$$\frac{\partial^2\boldsymbol{u}}{\partial t^2} = c_S^2\nabla^2\boldsymbol{u},$$

so each component of the displacement field in this case also propagates as a wave with speed c_S^2. The time it takes for such a wave to travel one characteristic length L is L/c_S, i.e., $L\sqrt{\varrho/\mu}$, which is nothing but our characteristic time t_c.

P waves. We may take the divergence of the PDE instead and notice that $\nabla\cdot\nabla = \nabla^2$ so

$$\frac{\partial^2}{\partial t^2}\nabla\cdot\boldsymbol{u} = c_P^2\nabla^2\nabla\cdot\boldsymbol{u},$$

with wave velocity

$$c_P = \sqrt{\frac{\lambda + 2\mu}{\varrho}}\,.$$

That is, the volume change (expansion/compression) propagates as a wave with speed c_P. These types of waves are called P waves. Other names are pressure and expansion/compression waves.

Suppose now that $\nabla \times \boldsymbol{u} = 0$, i.e., there is no rotation ("shear") of continuum elements. Mathematically this condition implies that $\nabla^2 \boldsymbol{u} = \nabla(\nabla \cdot \boldsymbol{u})$ (see any book on vector calculus or Wikipedia). Our model equation (4.5) then reduces to

$$\frac{\partial^2 \boldsymbol{u}}{\partial t^2} = c_P^2 \nabla^2 \boldsymbol{u},$$

which is nothing but a wave equation for the expansion component of the displacement field, just as (4.1.3) is for the shear component.

Time-varying load. Suppose we have some time-varying boundary condition on $\boldsymbol{u}$ or the stress vector (traction), with a time scale $1/\omega$ (some oscillating movement that goes like $\sin \omega t$, for instance). We choose $t_c = 1/\omega$. The scaling now leads to

$$\gamma \frac{\partial^2 \bar{\boldsymbol{u}}}{\partial \bar{t}^2} = \bar{\nabla}((\beta \bar{\lambda} + \bar{\mu}) \bar{\nabla} \cdot \bar{\boldsymbol{u}}) + \bar{\nabla} \cdot (\bar{\mu} \bar{\nabla} \bar{\boldsymbol{u}}) + \bar{\varrho} \bar{\boldsymbol{f}}\,.$$

where we have set

$$u_c = \mu_c^{-1} f_c L^2 \varrho_c,$$

as before, and γ is a new dimensionless number,

$$\gamma = \frac{\varrho_c L^2 \omega^2}{\mu_c} = \left(\frac{L\sqrt{\varrho_c/\mu_c}}{1/\omega} \right)^2 .$$

The last rewrite shows that $\sqrt{\gamma}$ is the ratio of the time scale for S waves and the time scale for the forced movement on the boundary. The acceleration term can therefore be neglected when $\gamma \ll 1$, i.e., when the time scale for movement on the boundary is much larger than the time it takes for the S waves to travel through the domain. Since the velocity of S waves in solids is very large and the time scale correspondingly small, $\gamma \ll 1$ is very often the case in applications involving structural analysis. Exercise 4.1 explores related models and asks for comparisons of time scales for waves and mechanical vibrations in structures.

4.1.4 The stationary elasticity problem

Scaling of the PDE. We now look at the stationary version of (4.1) where the $\varrho \boldsymbol{u}_{tt}$ term is removed. The first step in the scaling is just inserting the dimensionless variables:

$$0 = L^{-2}u_c\bar{\nabla}((\lambda_c\bar{\lambda} + \mu_c\bar{\mu})\bar{\nabla}\cdot\bar{\boldsymbol{u}}) + L^{-2}u_c\mu_c\bar{\nabla}\cdot(\bar{\mu}\bar{\nabla}\bar{\boldsymbol{u}}) + \varrho_c f_c\bar{\varrho}\bar{\boldsymbol{f}}\,.$$

Dividing by $L^2 u_c\mu_c$ gives

$$0 = \bar{\nabla}((\beta\bar{\lambda} + \bar{\mu})\bar{\nabla}\cdot\bar{\boldsymbol{u}}) + \bar{\nabla}\cdot(\bar{\mu}\bar{\nabla}\bar{\boldsymbol{u}}) + \frac{L^2\varrho_c f_c}{u_c\mu_c}\bar{\varrho}\bar{\boldsymbol{f}}\,.$$

Choosing $u_c = \varrho L^2 f_c/\mu_c$ leads to

$$\bar{\nabla}((\beta\bar{\lambda} + \bar{\mu})\bar{\nabla}\cdot\bar{\boldsymbol{u}}) + \bar{\nabla}\cdot(\bar{\mu}\bar{\nabla}\bar{\boldsymbol{u}}) + \bar{\varrho}\bar{\boldsymbol{f}} = 0\,. \tag{4.6}$$

A homogeneous material with constant λ, μ, and ϱ is an interesting case (this corresponds to $\mu_c = \mu$, $\lambda_c = \lambda$, $\varrho_c = \varrho$, $\bar{\varrho} = \bar{\lambda} = \bar{\mu} = 1$):

$$(1+\beta)\bar{\nabla}(\bar{\nabla}\cdot\bar{\boldsymbol{u}}) + \bar{\nabla}^2\bar{\boldsymbol{u}}) + \bar{\boldsymbol{f}} = 0\,. \tag{4.7}$$

Now β is defined as

$$\beta = \frac{\lambda}{\mu} = \left(\frac{c_p}{c_s}\right)^2 - 2\,.$$

It shows that in standard, stationary elasticity, λ/μ is the only significant physical parameter.

Remark on the characteristic displacement. The presented scaling may not be valid for problems where the geometry involves some dimensions that are much smaller than others, such as for beams, shells, or plates. Then one more length scale must be defined which gives us non-dimensional geometrical numbers. Global balances of moments and loads then determine how characteristic displacements depend on these numbers. As an example, consider a cantilever beam of length L and square-shaped cross section of width W, deformed under its own weight. From beam theory one can derive $u_c = \frac{3}{2}\varrho g L^2\delta^2/E$, where $\delta = L/W$ (g is the acceleration of gravity). If we consider E to be of the same size as λ, this implies that $\gamma \sim \delta^{-2}$. So, it may be wise to prescribe a u_c in elasticity problems, perhaps from formulas as shown, and keep γ in the PDE.

Scaling of displacement boundary conditions. A typical boundary condition on some part of the boundary is a prescribed displacement. For simplicity, we set $\boldsymbol{u} = \boldsymbol{U}_0$ for a constant vector $\boldsymbol{U}_0$ as boundary condition. With $u_c = \varrho L^2 f_c/\mu$, we get the dimensionless condition

$$\bar{\boldsymbol{u}} = \frac{\boldsymbol{U}_0}{u_c} = \frac{\mu \boldsymbol{U}_0}{\varrho L^2 f_c}\,.$$

In the absence of body forces, the expression for u_c has no meaning ($f_c = 0$), so then $u_c = |\boldsymbol{U}_0|$ is a better choice. This gives the dimensionless boundary condition

$$\bar{u} = \frac{\boldsymbol{U}_0}{|\boldsymbol{U}_0|},$$

which is the unit vector in the direction of $\boldsymbol{U}_0$. The new u_c changes the coefficient in front of the body force term, if that term is present, to the dimensionless number

$$\delta = \frac{L^2 \varrho f_c}{\mu |\boldsymbol{U}_0|}\,.$$

Scaling of traction boundary conditions. The other type of common boundary condition in elasticity is a prescribed traction (stress vector) on some part of the boundary:

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{T}_0,$$

where, to make it simple, we take $\boldsymbol{T}_0$ as a constant vector. From Section 4.1.2 we have a stress scale $\sigma_c = \mu u_c / L$, but we may alternatively use $|\boldsymbol{T}_0|$ as stress scale. In that case,

$$\bar{\boldsymbol{\sigma}} \cdot \boldsymbol{n} = \frac{\boldsymbol{T}_0}{|\boldsymbol{T}_0|},$$

which is a unit vector in the direction of $\boldsymbol{T}_0$. Many applications involve large traction free areas on the boundary, on which we simply have $\bar{\boldsymbol{\sigma}} \cdot \boldsymbol{n} = 0$.

4.1.5 Quasi-static thermo-elasticity

Heating solids gives rise to expansion, i.e., strains, which may cause stress if displacements are constrained. The time scale of temperature changes are usually much larger than the time scales of elastic waves, so the stationary equations of elasticity can be used, but a term depends on the temperature, so the equations must be coupled to a PDE for heat transfer in solids. The resulting system of PDEs is known as the equations of *thermo-elasticity* and reads

$$\nabla((\lambda+\mu)\nabla\cdot\boldsymbol{u})+\nabla\cdot(\mu\nabla\boldsymbol{u}) = \alpha\nabla T - \varrho\boldsymbol{f}, \tag{4.8}$$

$$\varrho c\frac{\partial T}{\partial t} = \nabla\cdot(\kappa\nabla T) + \varrho\boldsymbol{f}_T, \tag{4.9}$$

where T is the temperature, α is a coefficient of thermal expansion, c is a heat capacity, κ is the heat conduction coefficient, and $\boldsymbol{f}_T$ is some heat source. The density ϱ is strictly speaking a function of T and the stress state, but a widely used approximation is to consider ϱ as a constant. Most thermo-elasticity applications have $\boldsymbol{f}_T = 0$, so we drop this term. Most applications also involve some heating from a temperature level T_0 to some level $T_0 + \Delta T$. A suitable scaling for T is therefore

$$\bar{T} = \frac{T - T_0}{\Delta T},$$

so that $\bar{T} \in [0,1]$. The elasticity equation has already been scaled and so has the diffusion equation for T. We base the time scale on the diffusion, i.e., the thermal conduction process:

$$t_c = \varrho c L^2/\kappa_c\,.$$

We imagine that κ is scaled as $\bar{\kappa} = \kappa/\kappa_c$. The dimensionless PDE system then becomes

$$\bar{\nabla}((1+\beta)\bar{\mu}\bar{\nabla}\cdot\bar{\boldsymbol{u}})+\bar{\nabla}\cdot(\bar{\mu}\bar{\nabla}\bar{\boldsymbol{u}}) = \bar{\nabla}\bar{T} - \epsilon\bar{\varrho}\bar{\boldsymbol{f}}, \tag{4.10}$$

$$\frac{\partial\bar{T}}{\partial\bar{t}} = \bar{\nabla}\cdot(\bar{\kappa}\bar{\nabla}\bar{T})\,. \tag{4.11}$$

Here we have chosen u_c such that the "heating source term" has a unit coefficient, acknowledging that this thermal expansion balances the stress terms with $\bar{\boldsymbol{u}}$. The corresponding displacement scale is

$$u_c = \frac{\alpha L\Delta T}{\mu_c}\,.$$

The dimensionless number in the body force term is therefore

$$\epsilon = \frac{L\varrho_c f_c}{\alpha\Delta T},$$

which measures the ratio of the body force term and the "heating source term".

A homogeneous body with constant ϱ, λ, μ, c, and κ is common. The PDE system reduces in this case to

$$\bar{\nabla}((1+\beta)\bar{\nabla}\cdot\bar{\boldsymbol{u}})+\bar{\nabla}^2\bar{\boldsymbol{u}}) = \bar{\nabla}\bar{T}-\epsilon\bar{\boldsymbol{f}}, \tag{4.12}$$

$$\frac{\partial \bar{T}}{\partial \bar{t}} = \bar{\nabla}^2\bar{T}\,. \tag{4.13}$$

In the absence of body forces, β is again the key parameter.

The boundary conditions for thermo-elasticity consist of the conditions for elasticity and the conditions for diffusion. Scaling of such conditions are discussed in Section 3.2 and 4.1.4.

4.2 The Navier-Stokes equations

This section shows how to scale various versions of the equations governing incompressible viscous fluid flow. We start with the plain Navier-Stokes equations without body forces and progress with adding the gravity force and a free surface. We also look at scaling low Reynolds number flow and oscillating flows.

4.2.1 The momentum equation without body forces

The Navier-Stokes equations for incompressible viscous fluid flow, without body forces, take the form

$$\varrho\left(\frac{\partial \boldsymbol{u}}{\partial t}+\boldsymbol{u}\cdot\nabla\boldsymbol{u}\right) = -\nabla p+\mu\nabla^2\boldsymbol{u}, \tag{4.14}$$

$$\nabla\cdot\boldsymbol{u} = 0\,. \tag{4.15}$$

The primary unknowns are the velocity $\boldsymbol{u}$ and the pressure p. Moreover, ϱ is the fluid density, and μ is the dynamic viscosity.

Scaling. We start, as usual, by introducing a notation for dimensionless independent and dependent variables:

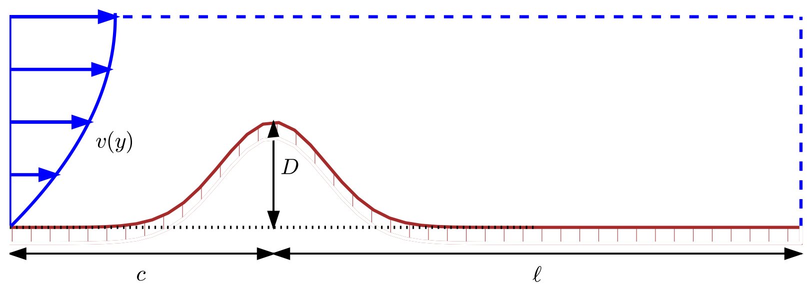
$$\bar{x} = \frac{x}{L}, \quad \bar{y} = \frac{y}{L}, \quad \bar{z} = \frac{z}{L}, \quad \bar{t} = \frac{t}{t_c}, \quad \bar{\boldsymbol{u}} = \frac{\boldsymbol{u}}{u_c}, \quad \bar{p} = \frac{p}{p_c},$$

where L is some characteristic distance, t_c is some characteristic time, u_c is a characteristic velocity, while p_c is a characteristic pressure. Inserted in the equations,

$$\varrho\left(\frac{u_c}{t_c}\frac{\partial\bar{\boldsymbol{u}}}{\partial\bar{t}}+\frac{u_c^2}{L}\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\boldsymbol{u}}\right)=-\frac{p_c}{L}\bar{\nabla}\bar{p}+\frac{u_c}{L^2}\mu\bar{\nabla}^2\bar{\boldsymbol{u}}, \tag{4.16}$$

$$\frac{u_c}{L}\bar{\nabla}\cdot\bar{\boldsymbol{u}}=0\,. \tag{4.17}$$

For the velocity it is common to just introduce some U for u_c. This U is normally implied by the problem description. For example, in the flow configuration below, with flow over a bump, we have some incoming flow with a profile $v(y)$ and U can typically be chosen as $U=\max_y v(y)$. The height of the bump influences the wake behind the bump, and is the length scale that really impacts the flow, so it is natural to set $L=D$. For numerical simulations in a domain of finite extent, $[0,c+\ell]$, c must be large enough to avoid feedback on the inlet profile, and ℓ must be large enough for the type of outflow boundary condition used. Ideally, $c,\ell\to\infty$, so none of these parameters are useful as length scales.



For flow in a channel or tube, we also have some inlet profile, e.g., $v(r)$ in a tube, where r is the radial coordinate. A natural choice of characteristic velocity is $U=v(0)$ or to let U be the average flow, i.e.,

$$U=\frac{1}{\pi R^2}\int_0^R 2\pi v(r)rdr,$$

if R is the radius of the tube. Other examples may be flow around a body, where there is some distant constant inlet flow $\boldsymbol{u}=U_0\boldsymbol{i}$, for instance, and $U=U_0$ is an obvious choice. We therefore assume that the flow problem itself brings a natural candidate for U.

Having a characteristic distance L and velocity U, an obvious time measure is L/U so we set $t_c=L/U$. Dividing by the coefficient in front of the time derivative term, creates a pressure term

$$\frac{p_c}{\varrho U^2}\bar{\nabla}\bar{p}\,.$$

The coefficient suggest a choice $p_c = \varrho U^2$ if the pressure gradient term is to have the same size as the acceleration terms. This p_c is a very common pressure scale in fluid mechanics, arising from Bernoulli's equation

$$p + \frac{1}{2}\varrho \boldsymbol{u} \cdot \boldsymbol{u} = \text{const}$$

for stationary flow.

Dimensonless PDEs and the Reynolds number. The discussions so far result in the following dimensionless form of (4.14) and (4.15):

$$\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \bar{\boldsymbol{u}} \cdot \bar{\nabla} \bar{\boldsymbol{u}} = -\bar{\nabla} \bar{p} + \mathrm{Re}^{-1} \bar{\nabla}^2 \bar{\boldsymbol{u}}, \tag{4.18}$$

$$\bar{\nabla} \cdot \bar{\boldsymbol{u}} = 0, \tag{4.19}$$

where Re is the famous *Reynolds number*,

$$\mathrm{Re} = \frac{\varrho U L}{\mu} = \frac{U L}{\nu}\,.$$

The latter expression makes use of the kinematic viscosity $\nu = \mu/\varrho$. For viscous fluid flows without body forces there is hence only one dimensionless number, Re.

The Reynolds number can be interpreted as the ratio of convection and viscosity:

$$\frac{\text{convection}}{\text{viscosity}} = \frac{|\varrho \boldsymbol{u} \cdot \nabla \boldsymbol{u}|}{|\mu \nabla^2 \boldsymbol{u}|} \sim \frac{\varrho U^2/L}{\mu U/L^2} = \frac{U L}{\nu} = \mathrm{Re}\,.$$

(We have here used that $\nabla \boldsymbol{u}$ goes like U/L and $\nabla^2 \boldsymbol{u}$ goes like U/L^2.)

4.2.2 Scaling of time for low Reynolds numbers

As we discussed in Section 3.4 for the convection-diffusion equation, there is not just one scaling that fits all problems. Above, we used $t_c = L/U$, which is appropriate if convection is a dominating physical effect. In case the convection term $\varrho \boldsymbol{u} \cdot \nabla \boldsymbol{u}$ is much smaller than the viscosity term $\mu \nabla^2 \boldsymbol{u}$, i.e., the Reynolds number is small, the viscosity term is dominating. However, if the scaling is right, the other terms are of order unity, and $\mathrm{Re}^{-1} \nabla^2 \bar{\boldsymbol{u}}$ must then also be of unit size. This fact implies that $\nabla^2 \bar{\boldsymbol{u}}$ must be small, but then the scaling is not right (since a right scaling will lead to $\bar{u}$ and its derivatives around unity). Such reasoning around inconsistent size of terms clearly points to the need for other scales.

In the low-Reynolds number regime, the diffusion effect of $\nabla^2 \bar{\boldsymbol{u}}$ is dominating, and we should use a time scale based on diffusion rather than convection.

Such a time scale is $t_c = L^2/(\mu/\varrho) = L^2/\nu$. With this time scale, the dimensionless Navier-Stokes equations look like

$$\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \mathrm{Re}\, \bar{\boldsymbol{u}} \cdot \bar{\nabla} \bar{\boldsymbol{u}} = -\bar{\nabla} p + \bar{\nabla}^2 \bar{\boldsymbol{u}}, \tag{4.20}$$

$$\bar{\nabla} \cdot \bar{\boldsymbol{u}} = 0\,. \tag{4.21}$$

As stated in the box in Section 3.4, (4.20) is the appropriate PDE for very low Reynolds number flow and suggests neglecting the convection term. If the flow is also steady, the time derivative term can be neglected, and we end up with the so-called *Stokes problem* for steady, slow, viscous flow:

$$-\bar{\nabla} p + \bar{\nabla}^2 \bar{\boldsymbol{u}} = 0, \tag{4.22}$$

$$\bar{\nabla} \cdot \bar{\boldsymbol{u}} = 0\,. \tag{4.23}$$

This flow regime is also known as *Stokes' flow* or *creeping flow*.

4.2.3 Shear stress as pressure scale

Instead of using the kinetic energy ϱU^2 as pressure scale, one can use the shear stress $\mu U/L$ (U/L reflects the spatial derivative of the velocity, which enters the shear stress expression $\mu \partial u/\partial y$). Using U as velocity scale, L/U as time scale, and $\mu U/L$ as pressure scale, results in

$$\mathrm{Re}\left(\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \bar{\boldsymbol{u}} \cdot \bar{\nabla} \bar{\boldsymbol{u}}\right) = -\bar{\nabla} \bar{p} + \bar{\nabla}^2 \bar{\boldsymbol{u}}\,. \tag{4.24}$$

Low Reynolds number flow now suggests neglecting both acceleration terms.

4.2.4 Gravity force and the Froude number

We now add a gravity force to the momentum equation (4.14):

$$\varrho\left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u}\right) = -\nabla p + \mu \nabla^2 \boldsymbol{u} - \varrho g \boldsymbol{k}, \tag{4.25}$$

where g is the acceleration of gravity, and $\boldsymbol{k}$ is a unit vector in the opposite direction of gravity. The new term takes the following form after non-dimensionalization:

$$\frac{t_c}{\varrho u_c}\varrho g\boldsymbol{k} = \frac{Lg}{U^2}\boldsymbol{k} = \mathrm{Fr}^{-2}\boldsymbol{k},$$

where Fr is the dimensionless Froude number,

$$\mathrm{Fr} = \frac{U}{\sqrt{Lg}}\,.$$

This quantity reflects the ratio of inertia and gravity forces:

$$\frac{|\boldsymbol{u}\cdot\nabla\boldsymbol{u}|}{|\varrho g|} \sim \frac{\varrho U^2/L}{\varrho g} = \mathrm{Fr}^2\,.$$

4.2.5 Oscillating boundary conditions and the Strouhal number

Many flows have an oscillating nature, often arising from some oscillating boundary condition. Suppose such a condition, at some boundary $x = \mathrm{const}$, takes the specific form

$$\boldsymbol{u} = U\sin(\omega t)\boldsymbol{i}\,.$$

The dimensionless counterpart becomes

$$U\bar{\boldsymbol{u}} = U\sin(\omega\frac{L}{U}\bar{t})\boldsymbol{i},$$

if $t_c = L/U$ is the appropriate time scale. This condition can be written

$$\bar{\boldsymbol{u}} = \sin(\mathrm{St}\,\bar{t}), \tag{4.26}$$

where St is the *Strouhal number*,

$$\mathrm{St} = \frac{\omega L}{U}\,. \tag{4.27}$$

The two important dimensionless parameters in oscillating flows are then the Reynolds and Strouhal numbers.

Even if the boundary conditions are of steady type, as for flow around a sphere or cylinder, the flow may at certain Reynolds numbers get unsteady and oscillating. For $10^2 < \mathrm{Re} < 10^7$, steady inflow towards a cylinder will cause vortex shedding: an array of vortices are periodically shedded from the cylinder, producing an oscillating flow pattern and force on the cylinder. The Strouhal number is used to characterize the frequency of oscillations. The phenomenon, known as *von Karman vortex street*, is particularly important if the frequency of the force on the cylinder hits the free vibration frequency of the cylinder such that resonance occurs. The result can be large displacements of the cylinder and structural failure. A famous case in engineering is the failure of the Tacoma Narrows suspension bridge in 1940, when wind-induced vortex shedding caused resonance with the free torsional vibrations of the bridge.

4.2.6 Cavitation and the Euler number

The dimensionless pressure in (4.18) made use of the pressure scale $p_c = \varrho U^2$. This is an appropriate scale if the pressure level is not of importance, which is very often the case since only the pressure *gradient* enters the flow equation and drives the flow. However, there are circumstances where the pressure level is of importance. For example, in some flows the pressure may become so low that the vapor pressure of the liquid is reached and that vapor cavities form (a phenomenon known as *cavitation*). A more appropriate pressure scale is then $p_c = p_\infty - p_v$, where p_∞ is a characteristic pressure level far from vapor cavities and p_v is the vapor pressure. The coefficient in front of the dimensionless pressure gradient is then

$$\frac{p_\infty - p_v}{\varrho U^2}\,.$$

Inspired by Bernoulli's equation $p + \frac{1}{2}\varrho \boldsymbol{u}\cdot\boldsymbol{u} = \text{const}$ in fluid mechanics, a factor $\frac{1}{2}$ is often inserted in the denominator. The corresponding dimensionless number,

$$\mathrm{Eu} = \frac{p_\infty - p_v}{\frac{1}{2}\varrho U^2}, \tag{4.28}$$

is called the *Euler number*. The pressure gradient term now reads $\frac{1}{2}\mathrm{Eu}\,\bar{\nabla}\bar{p}$. The Euler number expresses the ratio of pressure differences and the kinetic energy of the flow.

4.2.7 Free surface conditions and the Weber number

At a free surface, $z = \eta(x,y,t)$, the boundary conditions are

$$w = \frac{\partial \eta}{\partial t} + \boldsymbol{u}\cdot\nabla\eta, \tag{4.29}$$

$$p - p_0 \approx -\sigma\left(\frac{\partial^2\eta}{\partial x^2} + \frac{\partial^2\eta}{\partial y^2}\right), \tag{4.30}$$

where w is the velocity component in the z direction, p_0 is the atmospheric air pressure at the surface, and σ represents the surface tension. The approximation in (4.30) is valid under small deformations of the surface.

The dimensionless form of these conditions starts with inserting the dimensionless quantities in the equations:

$$u_c\bar{w} = \frac{L}{t_c}\frac{\partial\bar{\eta}}{\partial\bar{t}} + u_c\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\eta},$$

$$p_c\bar{p} \approx -\frac{1}{L}\sigma\left(\frac{\partial^2\bar{\eta}}{\partial\bar{x}^2} + \frac{\partial^2\bar{\eta}}{\partial\bar{y}^2}\right).$$

The characteristic length L is usually taken as the depth of the fluid when the surface is flat. We have used $\bar{p} = (p - p_0)/p_c$ for making the pressure dimensionless. Using $u_c = U$, $t_c = L/U$, and $p_c = \varrho U^2$, results in

$$\bar{w} = \frac{\partial\bar{\eta}}{\partial\bar{t}} + \bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\eta}, \tag{4.31}$$

$$\bar{p} \approx -\mathrm{We}^{-1}\left(\frac{\partial^2\bar{\eta}}{\partial\bar{x}^2} + \frac{\partial^2\bar{\eta}}{\partial\bar{y}^2}\right), \tag{4.32}$$

where We is the *Weber number*,

$$\mathrm{We} = \frac{\varrho U^2 L}{\sigma}. \tag{4.33}$$

The Weber number measures the importance of surface tension effects and is the ratio of the pressure scale ϱU^2 and the surface tension force per area, typically σ/R_x in a 2D problem, which has size σ/L.

4.3 Thermal convection

Temperature differences in fluid flow cause density differences, and since cold fluid is heavier than hot fluid, the gravity force will induce flow due to density differences. This effect is called free thermal convection and is key to our weather and heating of rooms. Forced convection refers to the case where there is no feedback from the temperature field to the motion, i.e., temperature differences do not create motion. This fact decouples the energy equation from the mass and momentum equations.

4.3.1 Forced convection

The model governing forced convection consists of the Navier-Stokes equations and the energy equation for the temperature:

$$\varrho\left(\frac{\partial \boldsymbol{u}}{\partial t}+\boldsymbol{u}\cdot\nabla\boldsymbol{u}\right)=-\nabla p+\mu\nabla^2\boldsymbol{u}-\varrho g\boldsymbol{k}, \tag{4.34}$$

$$\nabla\cdot\boldsymbol{u}=0, \tag{4.35}$$

$$\varrho c\left(\frac{\partial T}{\partial t}+\boldsymbol{u}\cdot\nabla T\right)=\kappa\nabla^2 T. \tag{4.36}$$

The symbol T is the temperature, c is a heat capacity, and κ is the heat conduction coefficient for the fluid. The PDE system applies primarily for liquids. For gases one may need a term $-p\nabla\cdot\boldsymbol{u}$ for the pressure work in (4.36) as well as a modified equation of continuity (4.35).

Despite the fact that ϱ depends on T, we treat ϱ as a constant ϱ_0. The major effect of the $\varrho(T)$ dependence is through the buoyancy effect caused by the gravity term $-\varrho(T)g\boldsymbol{k}$. It is common to drop this term in forced convection, and assume the momentum and continuity equations to be independent of the temperature. The flow is driven by boundary conditions (rather than density variations as in free convection), from which we can find a characteristic velocity U.

Dimensionless parameters are introduced as follows:

$$\bar{x}=\frac{x}{L},\ t_c=\frac{L}{U},\ \bar{\boldsymbol{u}}=\frac{\boldsymbol{u}}{U},\ \bar{p}=\frac{p}{\varrho_0 U^2},\ \bar{T}=\frac{T-T_0}{T_c}.$$

Other coordinates are also scaled by L. The characteristic temperature T_c is chosen as some range ΔT, which depends on the problem and is often given by the thermal initial and/or boundary conditions. The reference temperature T_0 is also implied by prescribed conditions. Inserted in the equations, we get

$$\varrho_0\frac{U^2}{L}\frac{\partial\bar{\boldsymbol{u}}}{\partial\bar{t}}+\varrho_0\frac{U^2}{L}\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\boldsymbol{u}}=-\frac{\varrho_0 U^2}{L}\bar{\nabla}\bar{p}+\frac{\mu U}{L^2}\bar{\nabla}^2\bar{\boldsymbol{u}},$$

$$\frac{U}{L}\bar{\nabla}\cdot\bar{\boldsymbol{u}}=0,$$

$$\varrho_0 c\left(\frac{T_c U}{L}\frac{\partial\bar{T}}{\partial\bar{t}}+\frac{UT_c}{L}\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{T}\right)=\frac{\kappa T_c}{L^2}\bar{\nabla}^2\bar{T}.$$

Making each term in each equation dimensionless reduces the system to

$$\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \bar{\boldsymbol{u}} \cdot \bar{\nabla} \bar{\boldsymbol{u}} = -\bar{\nabla} \bar{p} + \mathrm{Re}^{-1} \bar{\nabla}^2 \bar{\boldsymbol{u}}, \tag{4.37}$$

$$\bar{\nabla} \cdot \bar{\boldsymbol{u}} = 0, \tag{4.38}$$

$$\frac{\partial \bar{T}}{\partial \bar{t}} + \bar{\boldsymbol{u}} \cdot \bar{\nabla} \bar{T} = \mathrm{Pe}^{-1} \bar{\nabla}^2 \bar{T}. \tag{4.39}$$

The two dimensionless numbers in this system are given by

$$\mathrm{Pe} = \frac{\varrho_0 c U L}{\kappa}, \quad \mathrm{Re} = \frac{UL}{\nu} \quad (\nu = \frac{\mu}{\varrho_0}).$$

The Peclet number is here defined as the ratio of the convection term for heat $\varrho_0 c U \Delta T / L$ and the heat conduction term $\kappa U / L^2$. The fraction $\kappa / (\varrho_0 c)$ is known as the thermal diffusivity, and if this quantity is given a symbol α, we realize the relation to the Peclet number defined in Section 3.4.

4.3.2 Free convection

Governing equations. The mathematical model for free thermal convection consists of the Navier-Stokes equations coupled to an energy equation governing the temperature:

$$\varrho \left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} \right) = -\nabla p + \mu \nabla^2 \boldsymbol{u} - \varrho g \boldsymbol{k}, \tag{4.40}$$

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\varrho \boldsymbol{u}) = 0, \tag{4.41}$$

$$\varrho c \left(\frac{\partial T}{\partial t} + \boldsymbol{u} \cdot \nabla T \right) = \kappa \nabla^2 T + 2 \mu \varepsilon_{ij} \varepsilon_{ij}, \tag{4.42}$$

where Einstein's summation convention is implied for the $\varepsilon_{ij}\varepsilon_{ij}$ term. The symbol T is the temperature, c is a heat capacity, κ is the heat conduction coefficient for the fluid. In free convection, the gravity term $-\varrho(T) g \boldsymbol{k}$ is essential since the flow is driven by temperature differences and the fact that hot fluid rises while cold fluid falls.

For a slightly compressible gas flow a term $-p \nabla \cdot \boldsymbol{u}$ may be needed in (4.42.

Heating by viscous effects. We have also included heating of the fluid due to the work of viscous forces, represented by the term $2\mu\varepsilon_{ij}\varepsilon_{ij}$, where ε_{ij} is the strain-rate tensor in the flow, defined by

$$\varepsilon_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) = \frac{1}{2} (\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T),$$

where u_i is the velocity in direction of x_i ($i = 1,2,3$ measures the space directions). The term $2\mu\varepsilon_{ij}\varepsilon_{ij}$ is actually much more relevant for forced convection, but was left out in Section 4.3.1 for mathematical simplicity. Heating by the work of viscous forces is often a very small effect and can be neglected, although it plays a major role in forging and extrusion of metals where the viscosity is very large (such processes require large external forces to drive the flow). The reason for including the work by viscous forces under the heading of free convection, is that we want to scale a more complete, general mathematical model for mixed force and free convection, and arrive at dimensionless numbers that can tell if this extra term is important or not.

Relation between density and temperature. Equation (4.40) and has already been made dimensionless in the previous section. The major difference is now that ϱ is no longer a constant, but a function of T. The relationship between ϱ and T is often taken as linear,

$$\varrho = \varrho_0 - \varrho_0\beta(T - T_0),$$

where

$$\beta = -\frac{1}{\varrho}\left(\frac{\partial\varrho}{\partial t}\right)_p,$$

is known as the thermal expansion coefficient of the fluid, and ϱ_0 is a reference density when the temperature is at T_0.

The Boussinesq approximation. A very common approximation, called the *Boussinesq approximation*, is to neglect the density variations in all terms except the gravity term. This is a good approximation unless the change in ϱ is large. With the linear $\varrho(T)$ formula and the Boussinesq approximation, (4.40)-(4.42) take the form

$$\varrho_0\left(\frac{\partial\boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u}\right) = -\nabla p + \mu\nabla^2\boldsymbol{u} - (\varrho_0 - \varrho_0\beta(T - T_0))g\boldsymbol{k}, \tag{4.43}$$

$$\nabla\cdot\boldsymbol{u} = 0, \tag{4.44}$$

$$\varrho_0 c\left(\frac{\partial T}{\partial t} + \boldsymbol{u}\cdot\nabla T\right) = \kappa\nabla^2 T + 2\mu\varepsilon_{ij}\varepsilon_{ij}\,. \tag{4.45}$$

A good justification of the Boussinesq approximation is provided by Tritton [11, Ch. 13].

Scaling. Dimensionless variables are introduced as

$$\bar{x} = \frac{x}{L},\quad t_c = \frac{L}{U},\ \bar{\boldsymbol{u}} = \frac{\boldsymbol{u}}{U},\ \bar{p} = \frac{p}{\varrho U^2},\ \bar{T} = \frac{T - T_0}{\Delta T}\,.$$

The dimensionless y and z coordinates also make use of L as scale. As in forced convection, we assume the characteristic temperature level T_0 and the scale ΔT are given by thermal boundary and/or initial conditions. Contrary to Sections 4.2 and 4.3.1, U is now not given by the problem description, but implied by ΔT.

Replacing quantities with dimensions by their dimensionless counterparts results in

$$\begin{gathered}
\varrho_0 \frac{U^2}{L}\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \varrho_0 \frac{U^2}{L}\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\boldsymbol{u}} = -\frac{p_c}{L}\bar{\nabla}\bar{p} + \frac{\mu U}{L^2}\bar{\nabla}^2\bar{\boldsymbol{u}} - \varrho_0 g\boldsymbol{k} + \varrho_0\beta T_c\bar{T}g\boldsymbol{k},\\
\frac{U}{L}\bar{\nabla}\cdot\bar{\boldsymbol{u}} = 0,\\
\varrho_0 c\left(\frac{T_c U}{L}\frac{\partial \bar{T}}{\partial \bar{t}} + \frac{UT_c}{L}\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{T}\right) = \frac{\kappa T_c}{L^2}\bar{\nabla}^2\bar{T} + 2\frac{\mu U}{L}\bar{\varepsilon}_{ij}\bar{\varepsilon}_{ij}\,.
\end{gathered}$$

These equations reduce to

$$\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\boldsymbol{u}} = -\bar{\nabla}\bar{p} + \mathrm{Re}^{-1}\bar{\nabla}^2\bar{\boldsymbol{u}} - \mathrm{Fr}^{-2}\boldsymbol{k} + \gamma\bar{T}\boldsymbol{k}, \tag{4.46}$$

$$\bar{\nabla}\cdot\bar{\boldsymbol{u}} = 0, \tag{4.47}$$

$$\frac{\partial \bar{T}}{\partial \bar{t}} + \bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{T} = \mathrm{Pe}^{-1}\bar{\nabla}^2\bar{T} + 2\delta\bar{\varepsilon}_{ij}\bar{\varepsilon}_{ij}\,. \tag{4.48}$$

The dimensionless numbers, in addition to Re and Fr, are

$$\gamma = \frac{g\beta L\Delta T}{U^2}, \quad \mathrm{Pe}^{-1} = \frac{\kappa}{\varrho_0 cUL}, \quad \delta = \frac{\mu U}{L\varrho_0 c\Delta T}\,.$$

The γ number measures the ratio of thermal buoyancy and the convection term:

$$\gamma = \frac{\varrho_0 g\beta\Delta T}{\varrho_0 U^2/L} = \frac{g\beta L\Delta T}{U^2}\,.$$

The Pe parameter is the fraction of the convection term and the thermal diffusion term:

$$\frac{|\varrho_0\boldsymbol{u}\cdot\nabla T|}{|\kappa\nabla^2 T|} \sim \frac{\varrho_0 cU\Delta TL^{-1}}{\kappa L^{-2}\Delta T} = \frac{\varrho cUL}{\kappa} = \mathrm{Pe}\,.$$

The δ parameter is the ratio of the viscous dissipation term and the convection term:

$$\frac{|\mu\nabla^2\boldsymbol{u}|}{|\varrho_0 c\boldsymbol{u}\cdot\nabla T|} \sim \frac{\mu U^2/L^2}{\varrho_0 cU\Delta T/L} = \frac{\mu U}{L\varrho_0 c\Delta T} = \delta\,.$$

4.3.3 The Grashof, Prandtl, and Eckert numbers

The problem with the above dimensionless numbers is that they involve U, but U is implied by ΔT. Assuming that the convection term is much bigger than the viscous diffusion term, the momentum equation features a balance between the buoyancy term and the convection term:

$$|\varrho_0 \boldsymbol{u} \cdot \nabla \boldsymbol{u}| \sim \varrho_0 g \beta \Delta T\,.$$

Translating this similarity to scales,

$$\varrho_0 U^2/L \sim \varrho_0 g \beta \Delta T,$$

gives an U in terms of ΔT :

$$U = \sqrt{\beta L \Delta T}\,.$$

The Reynolds number with this U now becomes

$$\mathrm{Re}_T = \frac{UL}{\nu} = \frac{\sqrt{g\beta L^3 \Delta T}}{\nu^2} = \mathrm{Gr}^{1/2},$$

where Gr is the Grashof number in free thermal convection:

$$\mathrm{Gr} = \mathrm{Re}_T^2 = \frac{g\beta L^3 \Delta T}{\nu^2}\,.$$

The Grashof number replaces the Reynolds number in the scaled equations of free thermal convection. We shall soon look at its interpretations, which are not as straightforward as for the Reynolds and Peclet numbers.

The above choice of U in terms of ΔT results in γ equal to unity:

$$\gamma = \frac{g\beta L \Delta T}{U^2} = \frac{g\beta L \Delta T}{g\beta L \Delta T} = 1\,.$$

The Peclet number can also be rewritten as

$$\mathrm{Pe} = \frac{\varrho c U L}{\kappa} = \frac{\mu c}{\kappa}\frac{\varrho U L}{\mu} = \mathrm{Pr}\mathrm{Re}^{-1} = \mathrm{Pr}\mathrm{Re}_T^{-1},$$

where Pr is the Prandtl number, defined as

$$\mathrm{Pr} = \frac{\mu c}{\kappa}\,.$$

The Prandtl number is the ratio of the momentum diffusivity (kinematic viscosity) and the thermal diffusivity. Actually, more detailed analysis shows that Pr reflects the ratio of the thickness of the thermal and velocity boundary layers: when $\mathrm{Pr} = 1$, these layers coincide, while $\mathrm{Pr} \ll 1$ implies that the thermal layer is much thicker than the velocity boundary layer, and vice versa for $\mathrm{Pr} \gg 1$.

The δ parameter is in free convection replaced by a combination of the Eckert number (Ec) and the Reynolds number. We have that

$$\mathrm{Ec} = \frac{U^2}{c\Delta T} = \delta \mathrm{Re}_T,$$

and consequently

$$\delta = \mathrm{Ec}\mathrm{Re}_T^{-1} = \mathrm{Ec}\mathrm{Gr}^{-1/2}.$$

Writing

$$\mathrm{Ec} = \frac{\varrho_0 U^2}{\varrho_0 c\Delta T},$$

shows that the Eckert number can be interpreted as the ratio of the kinetic energy of the flow and the thermal energy.

We use Gr instead of Re_T in the momentum equations and also instead of Pe in the energy equation (recall that $\mathrm{Pe} = \mathrm{PrRe} = \mathrm{PrRe}_T = \mathrm{PrGr}^{-1/2}$). The resulting scaled system becomes

$$\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \bar{\boldsymbol{u}} \cdot \bar{\nabla} \bar{\boldsymbol{u}} = -\bar{\nabla}\bar{p} + \mathrm{Gr}^{-1/2}\bar{\nabla}^2 \bar{\boldsymbol{u}} - \mathrm{Fr}^{-2}\boldsymbol{k} + \bar{T}\boldsymbol{k}, \tag{4.49}$$

$$\bar{\nabla} \cdot \bar{\boldsymbol{u}} = 0, \tag{4.50}$$

$$\mathrm{Gr}^{1/2}\left(\frac{\partial \bar{T}}{\partial \bar{t}} + \bar{\boldsymbol{u}} \cdot \bar{\nabla}\bar{T}\right) = \mathrm{Pr}^{-1}\bar{\nabla}^2\bar{T} + 2\mathrm{Ec}\mathrm{Gr}^{-1/2}\bar{\varepsilon}_{ij}\bar{\varepsilon}_{ij}\,. \tag{4.51}$$

The Grashof number plays the same role as the Reynolds number in the momentum equation in free convection. In particular, it turns out that Gr governs the transition between laminar and turbulent flow. For example, the transition to turbulence occurs in the range $10^8 < \mathrm{Gr} < 10^9$ for free convection from vertical flat plates. Gr is normally interpreted as a dimensionless number expressing the ratio of buoyancy forces and viscous forces.

Interpretations of the Grashof number. Recall that the scaling leading to the Grashof number is based on an estimate of U from a balance of the convective and the buoyancy terms. When the viscous term dominates over convection, we need a different estimate of U, since in this case, the viscous force balances the buoyancy force:

$$\mu\nabla^2\boldsymbol{u} \sim \varrho_0 g\beta\Delta T \quad \Rightarrow \quad \mu U/L^2 \sim \varrho_0 g\beta\Delta T,$$

This similarity suggests the scale

$$U = \frac{g\beta L^2\Delta T}{\nu}\,.$$

Now,

$$\frac{|\varrho_0 \boldsymbol{u}\cdot\nabla\boldsymbol{u}|}{|\mu\nabla^2\boldsymbol{u}|} \sim \frac{UL}{\nu} = \frac{g\beta L^3 \Delta T}{\nu} = \mathrm{Gr}\,.$$

The result means that $\mathrm{Gr}^{1/2}$ measures the ratio of convection and viscous forces when convection dominates, whereas Gr measures this ratio when viscous forces dominate.

The product of Gr and Pr is the Rayleigh number,

$$\mathrm{Ra} = \frac{g\beta L^3 \Delta T \varrho_0 c}{\nu\kappa},$$

since

$$\mathrm{GrPr} = \mathrm{Re}_T^2\mathrm{Pr} = \frac{g\beta L^3 \Delta T}{\nu^2}\frac{\mu c}{\kappa} = \frac{g\beta L^3 \Delta T \varrho_0 c}{\nu\kappa} = \mathrm{Ra}\,.$$

The Rayleigh number is the preferred dimensionless number when studying free convection in horizontal layers [2, 11]. Otherwise, Gr and Pr are used.

4.3.4 Heat transfer at boundaries and the Nusselt and Biot numbers

A common boundary condition, modeling heat transfer to/from the surroundings, is

$$-\kappa\frac{\partial T}{\partial n} = h(T - T_s), \tag{4.52}$$

where $\partial/\partial n$ means the derivative in the normal direction ($\boldsymbol{n}\cdot\nabla$), h is an experimentally determined heat transfer coefficient, and T_s is the temperature of the surroundings. Scaling (4.52) leads to

$$-\frac{\kappa\Delta t}{L}\frac{\partial\bar{T}}{\partial\bar{n}} = h(\Delta T\bar{T} + T_0 - T_s),$$

and further to

$$\frac{\partial\bar{T}}{\partial\bar{n}} = \frac{hL}{\kappa}(\bar{T} + \frac{T_s - T_0}{\Delta T}) = \delta(\bar{T} - \bar{T}_s),$$

where the dimensionless number δ is defined by

$$\delta = \frac{hL}{\kappa},$$

and $\bar{T}_s$ is simply the dimensionless surrounding temperature,

$$\bar{T}_s = \frac{T_s - T_0}{\Delta T}\,.$$

When studying heat transfer in a fluid, with solid surroundings, δ is known as the Nusselt number Nu. The left-hand side of (4.52) represents in this case heat conduction, while the right-hand side models convective heat transfer at a boundary. The Nusselt number can then be interpreted as the ratio of convective and conductive heat transfer at a solid boundary:

$$\frac{|h(T - T_s)|}{\kappa T/L} \sim \frac{h}{\kappa/L} = \mathrm{Nu}\,.$$

The case with heat transfer in a solid with a fluid as surroundings gives the same dimensionless δ, but the number is now known as the Biot number. It describes the ratio of heat loss/gain with the surroundings at the solid body's boundary and conduction inside the body. A small Biot number indicates that conduction is a fast process and one can use Newton's law of cooling (see Section 2.1.7) instead of a detailed calculation of the spatio-temporal temperature variation in the body. The Biot number also arises in simplified models of heat conduction in solids, see Exercise 3.5.

Heat transfer is a huge engineering field with lots of experimental investigations that are summarized by curves relating various dimensionless numbers such as Gr, Pr, Nu, and Bi.

4.4 Compressible gas dynamics

4.4.1 The Euler equations of gas dynamics

The fundamental equations for a compressible fluid are based on balance of mass, momentum, and energy. When molecular diffusion effects are negligible, the PDE system, known as the Euler equations of gas dynamics, can be written as

$$\frac{\partial \varrho}{\partial t} + \nabla\cdot(\varrho \boldsymbol{u}) = 0, \tag{4.53}$$

$$\frac{\partial (\varrho \boldsymbol{u})}{\partial t} + \nabla\cdot(\varrho \boldsymbol{u}\boldsymbol{u}^T) = -\nabla p + \varrho \boldsymbol{f}, \tag{4.54}$$

$$\frac{\partial E}{\partial t} + \nabla\cdot(\boldsymbol{u}(E + p)) = 0, \tag{4.55}$$

with E being

$$E = \varrho e + \frac{1}{2}\varrho \boldsymbol{u}\cdot\boldsymbol{u}\,. \tag{4.56}$$

In these equations, $\boldsymbol{u}$ is the fluid velocity, ϱ is the density, p is the pressure, E is the total energy per unit volume, composed of the kinetic energy per unit volume, $\frac{1}{2}\varrho\boldsymbol{u}\cdot\boldsymbol{u}$, and the internal energy per unit volume, ϱe.

Assuming the fluid to be an ideal gas implies the following additional relations:

$$e = c_v T, \tag{4.57}$$

$$p = \varrho RT = \frac{R}{c_v}(E - \frac{1}{2}\varrho\boldsymbol{u}\cdot\boldsymbol{u}), \tag{4.58}$$

where c_v is the specific heat capacity at constant volume (for dry air $c_v = 717.5\,\mathrm{J\,kg^{-1}K^{-1}}$), R is the specific ideal gas constant ($R = 287.14\mathrm{Jkg^{-1}K^{-1}}$), and T is the temperature.

The common way to solve these equations is to propagate ϱ, $\varrho\boldsymbol{u}$, and E by an explicit numerical method in time for (4.53)-(4.55), using (4.58) for p.

We introduce dimensionless independent variables,

$$\bar{x} = \frac{x}{L}, \quad \bar{y} = \frac{y}{L}, \quad \bar{z} = \frac{z}{L}, \quad \bar{t} = \frac{t}{t_c},$$

and dimensionless dependent variables,

$$\bar{\boldsymbol{u}} = \frac{\boldsymbol{u}}{U}, \quad \bar{\varrho} = \frac{\varrho}{\varrho_c}, \quad \bar{p} = \frac{p}{p_c}, \quad \bar{E} = \frac{E}{E_c}\,.$$

Inserting these expressions in the governing equations gives

$$\begin{aligned}
\frac{\partial\bar{\varrho}}{\partial\bar{t}} + \frac{t_c U}{L}\bar{\nabla}\cdot(\bar{\varrho}\bar{\boldsymbol{u}}) &= 0,\\
\frac{\partial(\bar{\varrho}\bar{\boldsymbol{u}})}{\partial\bar{t}} + \frac{t_c U}{L}\bar{\nabla}\cdot(\bar{\varrho}\bar{\boldsymbol{u}}\bar{\boldsymbol{u}}^T) &= -\frac{t_c p_c}{UL\varrho_c}\nabla\bar{p} + \frac{t_c f_c}{U}\bar{\varrho}\bar{\boldsymbol{f}},\\
\frac{\partial\bar{E}}{\partial\bar{t}} + \frac{t_c U}{LE_c}\bar{\nabla}\cdot(\bar{\boldsymbol{u}}(E_c\bar{E} + p_c\bar{p})) &= 0,\\
\bar{p} &= \frac{R}{c_v p_c}(E_c\bar{E} - \frac{1}{2}\varrho_c u_c\bar{\varrho}\bar{\boldsymbol{u}}\cdot\bar{\boldsymbol{u}})\,.
\end{aligned}$$

A natural choice of time scale is $t_c = L/U$. A common choice of pressure scale is $p_c = \varrho_c U^2$. The energy equation simplifies if we choose $E_c = p_c = \varrho_c U^2$. With these scales we get

$$
\begin{aligned}
\frac{\partial \bar{\varrho}}{\partial \bar{t}} + \bar{\nabla} \cdot (\bar{\varrho}\bar{\boldsymbol{u}}) &= 0, \\
\frac{\partial (\bar{\varrho}\bar{\boldsymbol{u}})}{\partial \bar{t}} + \bar{\nabla} \cdot (\bar{\varrho}\bar{\boldsymbol{u}}\bar{\boldsymbol{u}}^T) &= -\nabla \bar{p} + \alpha \bar{\varrho} \bar{\boldsymbol{f}}, \\
\frac{\partial \bar{E}}{\partial \bar{t}} + \bar{\nabla} \cdot (\bar{\boldsymbol{u}}(\bar{E} + \bar{p})) &= 0, \\
\bar{p} = \frac{R}{c_v}(\bar{E} - \frac{1}{2}\bar{\varrho}\bar{u} \cdot \bar{u}),
\end{aligned}
$$

where α is a dimensionless number:

$$\alpha = \frac{L f_c}{U^2}\,.$$

We realize that the scaled Euler equations look like the ones with dimension, apart from the α coefficient.

4.4.2 General isentropic flow

Heat transfer can be neglected in isentropic flow, and there is hence an equation of state involving only ϱ and p:

$$p = F(\varrho)\,.$$

The energy equation is now not needed and the Euler equations simplify to

$$\frac{\partial \varrho}{\partial t} + \nabla \cdot (\boldsymbol{u}\varrho) = 0, \tag{4.59}$$

$$\varrho \frac{\partial \boldsymbol{u}}{\partial t} + \varrho \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \nabla p = 0\,. \tag{4.60}$$

Elimination of the pressure. A common equation of state is

$$F(\varrho) = p_0 \left(\frac{\varrho}{\varrho_0}\right)^{\gamma},$$

where $\gamma = 5/3$ for air. The first step is to eliminate p in favor of ϱ so we get a system for ϱ and $\boldsymbol{u}$. To this end, we must calculate ∇p:

$$\nabla p = F'(\varrho)\nabla \varrho, \quad F'(\varrho) = c_0^2 \left(\frac{\varrho}{\varrho_0}\right)^{\gamma - 1},$$

where

$$c_0 = \sqrt{\frac{\gamma p_0}{\varrho_0}}$$

is the speed of sound within the fluid in the equilibrium state (see the subsequent section). Equation (4.60) with eliminated pressure p reads

$$\varrho\frac{\partial \boldsymbol{u}}{\partial t} + \varrho\boldsymbol{u}\cdot\nabla\boldsymbol{u} + c_0^2\left(\frac{\varrho}{\varrho_0}\right)^{\gamma-1}\nabla\varrho = 0\,. \tag{4.61}$$

The governing equations are now (4.59) and (4.61). Space and time are scaled in the usual way as

$$\bar{x} = \frac{x}{L},\quad \bar{y} = \frac{y}{L},\quad \bar{z} = \frac{z}{L},\quad \bar{t} = \frac{t}{t_c}\,.$$

The scaled dependent variables are

$$\bar{\varrho} = \frac{\varrho}{\varrho_c},\quad \bar{\boldsymbol{u}} = \frac{\boldsymbol{u}}{U}\,.$$

Then $F'(\varrho) = c_0^2\bar{\varrho}^{\gamma-1}$.

Inserting the dimensionless variables in the two governing PDEs leads to

$$\begin{aligned}\frac{\varrho_c}{t_c}\frac{\partial\bar{\varrho}}{\partial\bar{t}} + \frac{\varrho_c U}{L}\bar{\nabla}\cdot(\bar{\varrho}\bar{\boldsymbol{u}}) &= 0,\\ \frac{\varrho_c U}{t_c}\bar{\varrho}\frac{\partial\bar{\boldsymbol{u}}}{\partial\bar{t}} + \frac{\varrho_c U^2}{L}\bar{\varrho}\bar{u}\cdot\bar{\nabla}\bar{\boldsymbol{u}} + \frac{\varrho_c}{L}\left(\frac{\varrho_c}{\varrho_0}\right)^{\gamma-1} c_0^2\bar{\varrho}^{\gamma-1}\bar{\nabla}\bar{\varrho} &= 0\,.\end{aligned}$$

The characteristic flow velocity is U so a natural time scale is $t_c = L/U$. This choice leads to the scaled PDEs

$$\frac{\partial\bar{\varrho}}{\partial\bar{t}} + \bar{\nabla}\cdot(\bar{\varrho}\bar{\boldsymbol{u}}) = 0, \tag{4.62}$$

$$\bar{\varrho}\frac{\partial\bar{\boldsymbol{u}}}{\partial\bar{t}} + \bar{\varrho}\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\boldsymbol{u}} + \mathrm{M}^{-2}\left(\frac{\varrho_c}{\varrho_0}\right)^{\gamma-1}\bar{\varrho}^{\gamma-1}\bar{\nabla}\bar{\varrho} = 0, \tag{4.63}$$

where the dimensionless number

$$\mathrm{M} = \frac{U}{c_0},$$

is known as the *Mach number*. The boundary conditions specify the characteristic velocity U and thereby the Mach number. Observe that (4.63) simplifies if $\varrho_c = \varrho_0$ is an appropriate choice.

4.4.3 The acoustic approximation for sound waves

Wave nature of isentropic flow with small perturbations. A model for sound waves can be based on (4.59) and (4.61), but in this case there are small pressure, velocity, and density *perturbations* from a ground state at rest where $\boldsymbol{u}=0$, $\varrho=\varrho_0$, and $p=p_0=F(\varrho_0)$. Introducing the perturbations $\hat{\varrho}=\varrho-\varrho_0$ and $\hat{\boldsymbol{u}}$, (4.59) and (4.61) take the form

$$\begin{aligned}\frac{\partial\hat{\varrho}}{\partial t}+\nabla\cdot(\hat{\boldsymbol{u}}(\varrho_0+\hat{\varrho})&=0,\\(\varrho_0+\hat{\varrho})\frac{\partial\hat{\boldsymbol{u}}}{\partial t}+(\varrho_0+\hat{\varrho})\hat{\boldsymbol{u}}\cdot\nabla\hat{\boldsymbol{u}}+c_0^2\left(1+\frac{\hat{\varrho}}{\varrho_0}\right)^{\gamma-1}\nabla\hat{\varrho}&=0\,.\end{aligned}$$

For small perturbations we can linearize this PDE system by neglecting all products of $\hat{\varrho}$ and $\hat{\boldsymbol{u}}$. Also, $1+\hat{\varrho}/\varrho_0\approx 1$. This leaves us with the simplified system

$$\begin{aligned}\frac{\partial\hat{\varrho}}{\partial t}+\varrho_0\nabla\cdot\hat{\boldsymbol{u}}&=0,\\\varrho_0\frac{\partial\hat{\boldsymbol{u}}}{\partial t}+c_0^2\nabla\hat{\varrho}&=0\,.\end{aligned}$$

Eliminating $\hat{\boldsymbol{u}}$ by differentiating the first PDE with respect to t and taking the divergence of the second PDE gives a standard wave equation for the density perturbations:

$$\frac{\partial^2\hat{\varrho}}{\partial t^2}=c_0^2\nabla^2\hat{\varrho}\,.$$

Similarly, $\hat{\varrho}$ can be eliminated and one gets a wave equation for $\hat{\boldsymbol{u}}$, also with wave velocity c_0. This means that the sound perturbations travel with velocity c_0.

Basic scaling for small wave perturbations. Let ϱ_c and u_c be characteristic sizes of the perturbations in density and velocity. The density will then vary in $[\varrho_0-\varrho_c,\varrho_0+\varrho_c]$. An appropriate scaling is

$$\bar{\varrho}=\frac{\varrho-\varrho_0}{\varrho_c}$$

such that $\bar{\varrho}\in[-1,1]$. Consequently,

$$\varrho=\varrho_0+\varrho_c\bar{\varrho}=\varrho_0(1+\alpha\bar{\varrho}),\quad \alpha=\frac{\varrho_c}{\varrho_0}\,.$$

Note that the dimensionless α is expected to be a very small number since $\varrho_c\ll\varrho_0$. The velocity, space, and time are scaled as in the previous section. Also note that ϱ_0 and p_0 are known values, but the scales ϱ_c and U are not known. Usually these can be estimated from perturbations (i.e., sound generation) applied at the boundary.

Inserting the scaled variables in (4.59) and (4.61) results in

$$
\begin{aligned}
\alpha \frac{\varrho_0}{t_c} \frac{\partial \bar{\varrho}}{\partial \bar{t}} + \frac{\varrho_0 U}{L} \bar{\nabla} \cdot ((1+\alpha\bar{\varrho})\bar{\boldsymbol{u}}) = 0, \\
\frac{\varrho_0 U}{t_c}(1+\alpha\bar{\varrho})\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \frac{\varrho_0 U^2}{L}(1+\alpha\bar{\varrho})\bar{u}\cdot\bar{\nabla}\bar{\boldsymbol{u}} + \alpha\frac{\varrho_0}{L}c_0^2\,(1+\alpha\bar{\varrho})^{\gamma-1}\,\bar{\nabla}\bar{\varrho} = 0\,.
\end{aligned}
$$

Since we now model sound waves, the relevant time scale is not L/U but the time it takes a wave to travel through the domain: $t_c = L/c_0$. This is a much smaller time scale than in the previous section because $c_0 \gg U$ (think of humans speaking: the sound travels very fast but one cannot feel the corresponding very small flow perturbation in the air!). Using $t_c = L/u_0$ we get

$$
\begin{aligned}
\alpha \frac{\partial \bar{\varrho}}{\partial \bar{t}} + \mathrm{M}\bar{\nabla} \cdot ((1+\alpha\bar{\varrho})\bar{\boldsymbol{u}}) = 0, \\
(1+\alpha\bar{\varrho})\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \mathrm{M}(1+\alpha\bar{\varrho})\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\boldsymbol{u}} + \alpha\mathrm{M}^{-1}\,(1+\alpha\bar{\varrho})^{\gamma-1}\,\bar{\nabla}\bar{\varrho} = 0\,.
\end{aligned}
$$

For small perturbations the linear terms in these equations must balance. This points to M and α being of the same order and we may choose $\alpha = M$ (implying $\varrho_c = \varrho_0\mathrm{M}$) to obtain

$$
\begin{aligned}
\frac{\partial \bar{\varrho}}{\partial \bar{t}} + \bar{\nabla} \cdot ((1+\mathrm{M}\bar{\varrho})\bar{\boldsymbol{u}}) = 0, \\
\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \mathrm{M}\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\boldsymbol{u}} + (1+\mathrm{M}\bar{\varrho})^{\gamma-2}\,\bar{\nabla}\bar{\varrho} = 0\,.
\end{aligned}
$$

Now the Mach number, M, appears in the nonlinear terms only. Letting $\mathrm{M} \to 0$ we arrive at the following linearized system of PDEs

$$
\frac{\partial \bar{\varrho}}{\partial \bar{t}} + \bar{\nabla} \cdot \bar{\boldsymbol{u}} = 0, \tag{4.64}
$$

$$
\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \bar{\nabla}\bar{\varrho} = 0, \tag{4.65}
$$

The velocity $\boldsymbol{u}$ can be eliminated taking the time derivative of (4.64) and the divergence of (4.65):

$$
\frac{\partial^2 \bar{\varrho}}{\partial \bar{t}^2} = \bar{\nabla}^2 \bar{\varrho}, \tag{4.66}
$$

which is nothing but a standard dimensionless wave equation with unit wave velocity. Similarly, we can eliminate ϱ by taking the divergence of (4.64) and the time derivative of (4.65):

$$\frac{\partial^2 \bar{\boldsymbol{u}}}{\partial \bar{t}^2} = \bar{\nabla}^2 \bar{\boldsymbol{u}}\,. \tag{4.67}$$

We also observe that there are no physical parameters in the scaled wave equations.

4.5 Water surface waves driven by gravity

4.5.1 The mathematical model

Provided the Weber number (see section 4.2.7) is sufficiently small, capillary effects may be omitted and water surface waves are governed by gravity. For large Reynolds numbers, viscous effects may also be ignored (except in boundary layers close to the bottom or the surface of the fluid). The flow of an incompressible homogeneous fluid under these assumptions is governed by the Euler equations of motion on the form

$$\nabla \cdot \boldsymbol{u} = 0, \tag{4.68}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \frac{1}{\rho} \nabla p + g\mathbf{k} = 0\,. \tag{4.69}$$

When the free surface position is described as $z = \eta(x,y,t)$, with z as the vertical coordinate, the boundary conditions at the surface read

$$p = p_s, \tag{4.70}$$

$$\frac{\partial \eta}{\partial t} + \boldsymbol{u} \cdot \nabla \eta = w, \tag{4.71}$$

where p_s is the external pressure applied to the surface. At the bottom, $z = -h(x,y)$, there is the no-flux condition

$$\frac{\partial h}{\partial x} u + \frac{\partial h}{\partial y} v = -w\,.$$

In addition to ρ and g we assume that a typical depth h_c, a typical wavelength λ_c, and a typical surface elevation A, which then by definition is a scale for η, are the given parameters. From these we must derive scales for the coordinates, the velocity components, and the pressure.

4.5.2 Scaling

First, it is instructive to define a typical wave celerity, c_c, which must be linked to the length and time scale according to $c_c = \lambda_c/t_c$. Since there is no other given parameter that matches the mass dimension of ρ, we express c_c in terms of A, λ_c, h_c, and g. Most of the work on waves in any discipline of physics is devoted to linear or weakly nonlinear waves, and the wave celerity must be presumed to remain finite as A goes to zero (see, for instance, Section 4.4.3). Hence, we may assume that c_c must depend on g and either h_c or λ_c. Next, the two horizontal directions are equivalent with regard to scaling, implying that we have a common velocity scale, U, for u and v, a common length scale L for x and y. The obvious choice for L is λ_c, while the "vertical quantities" w and z have scales W and Z, respectively, which may differ from the horizontal counterparts. However, we assume that also the length scale Z remains finite as $A \to 0$ and hence is independent of A. This is less obvious for Z than for c_c and t_c, but may eventually be confirmed by the existence of linear solutions when solving the equation set. From the linear part of (4.71) and (4.68) we obtain two relations between velocity and coordinate scales by demanding the non-dimensionalized terms to be of order unity

$$\frac{A}{t_c} = W, \quad \frac{U}{L} = \frac{W}{Z}\,. \tag{4.72}$$

These relations are indeed useful, but they do not suffice to establish the scaling.

The pressure may be regarded as the sum of a large equilibrium part, balancing gravity, and a much smaller dynamic part associated with the presence of waves. To make the latter appear in the equations we define the dynamic pressure, p_d, according to

$$p = p_s - \rho g z + p_d,$$

and the pressure scale $p_c = \rho g A$ for p_d then follows directly from the surface condition (4.70).

The equation set will be scaled according to

$$\bar{t} = \frac{t}{t_c},\ \bar{x} = \frac{x}{L},\ \bar{y} = \frac{y}{L},\ \bar{z} = \frac{z}{Z},\ \bar{\eta} = \frac{\eta}{A},\ \bar{u} = \frac{u}{U},\ \bar{v} = \frac{v}{U},\ \bar{w} = \frac{w}{W},\ \bar{p}_d = \frac{p_d}{p_c}\,.$$

In the further development of the scaling we focus on two limiting cases, namely deep and shallow water.

4.5.3 Waves in deep water

Deep water means that $h_c \gg \lambda_c$. Presumably the waves will not feel the bottom, and h as well as h_c are removed from our equations. The bottom boundary condition is replaced by a requirement of vanishing velocity as $z \to -\infty$. Consequently, c_c must depend upon λ_c and g, leaving us with $c_c = \sqrt{g\lambda_c}$ and $Z = \lambda_c = L$ as the only options. Then, $t_c = \sqrt{\lambda_c/g}$ and (4.72) implies $U = W = c_0 \frac{A}{\lambda_c} = \epsilon c_0$, where we have introduced the non-dimensional number

$$\epsilon = \frac{A}{\lambda_c},$$

which is the wave steepness. The equality of the horizontal and the vertical scale is consistent with the common knowledge that the particle orbits in deep water surface waves are circular.

The scaled equations are now expressed with ϵ as sole dimensionless number

$$\bar{\nabla} \cdot \bar{\boldsymbol{u}} = 0, \tag{4.73}$$

$$\frac{\partial \bar{\boldsymbol{u}}}{\partial \bar{t}} + \epsilon \bar{\boldsymbol{u}} \cdot \bar{\nabla} \bar{\boldsymbol{u}} + \bar{\nabla} \bar{p}_d = 0\,. \tag{4.74}$$

The surface conditions, at $z = \epsilon\eta$, become

$$\bar{p}_d = \bar{\eta}, \tag{4.75}$$

$$\frac{\partial \bar{\eta}}{\partial \bar{t}} + \epsilon \bar{\boldsymbol{u}} \cdot \bar{\nabla} \bar{\eta} = \bar{w}, \tag{4.76}$$

while the bottom condition is replaced by

$$\bar{\boldsymbol{u}} \to 0, \tag{4.77}$$

as $\bar{z} \to -\infty$.

4.5.4 Long waves in shallow water

Long waves imply that the wavelength is large compared to the depth: $\lambda_c \gg h_c$. In analogy with the reasoning above, we presume that the speed of the waves remains finite as $\lambda_c \to \infty$. Then, c_c must be based on g and h_c, which leads to $c_c = \sqrt{gh_c}$ and $t_c = \lambda_c/\sqrt{gh_c}$. The natural choice for the vertical length scale is now the depth; $Z = h_c$. Application of (4.72) then leads to $W = c_c A/\lambda_c$ and $U = c_c A/h_c$.

Introducing the dimensionless numbers

$$\alpha = \frac{A}{h_c}, \quad \mu = \frac{h_c}{\lambda_c},$$

we rewrite the velocity scales as

$$W = \mu\alpha c_c, \quad U = \alpha c_c\,.$$

We observe that $W \ll U$ for shallow water and that particle orbits must be elongated in the horizontal direction.

The equation set is now most transparently written by introducing the horizontal velocity $\bar{\boldsymbol{u}}_h = \bar{u}\mathbf{i} + \bar{v}\mathbf{j}$ and the corresponding horizontal components of the gradient operator, $\bar{\nabla}_h$:

$$\bar{\nabla}\cdot\bar{\boldsymbol{u}}_h + \frac{\partial\bar{w}}{\partial\bar{z}} = 0, \tag{4.78}$$

$$\frac{\partial\bar{\boldsymbol{u}}_h}{\partial\bar{t}} + \alpha\bar{\boldsymbol{u}}_h\cdot\bar{\nabla}_h\bar{\boldsymbol{u}}_h + \alpha\bar{w}\frac{\partial\bar{\boldsymbol{u}}_h}{\partial\bar{z}} + \bar{\nabla}_h\bar{p}_d = 0, \tag{4.79}$$

$$\mu^2\left(\frac{\partial\bar{\boldsymbol{w}}}{\partial\bar{t}} + \alpha\bar{\boldsymbol{u}}_h\cdot\bar{\nabla}_h\bar{\boldsymbol{w}} + \alpha\bar{w}\frac{\partial\bar{w}}{\partial\bar{z}}\right) + \frac{\partial\bar{p}_d}{\partial\bar{z}} = 0.. \tag{4.80}$$

Surface conditions, at $z = \alpha\eta$, now become

$$\bar{p}_d = \bar{\eta}, \tag{4.81}$$

$$\frac{\partial\bar{\eta}}{\partial\bar{t}} + \alpha\bar{\boldsymbol{u}}_h\cdot\bar{\nabla}_h\bar{\eta} = \bar{w}, \tag{4.82}$$

while the bottom condition is invariant with respect to the present scaling

$$\bar{\nabla}_h\cdot\bar{\boldsymbol{u}}_h = -\bar{w}\,. \tag{4.83}$$

An immediate consequence is that $\bar{p}_d$ remains equal to $\bar{\eta}$ throughout the water column when $\mu^2 \to 0$, which implies that the pressure is hydrostatic. The above set of equations is a common starting point for perturbation expansions in ϵ and μ^2 that lead to shallow water, KdV, and Boussinesq type equations.

4.6 Two-phase porous media flow

We consider the flow of two incompressible, immiscible fluids in a porous medium with porosity $\phi(\boldsymbol{x})$. The two fluids are referred to as the wetting and non-wetting fluid. In an oil-water mixture, water is usually the wetting fluid. The fraction of the pore volume occupied by the wetting fluid is denoted by

$S(\boldsymbol{x},t)$. The non-wetting fluid then occupies $1-S$ of the pore volume (or $(1-S)\phi$ of the total volume). The variable $P(\boldsymbol{x},t)$ represents the pressure in the non-wetting fluid. It is related to the pressure P_n in the non-wetting fluid through the capillary pressure $p_c = P_n - P$, which is an empirically determined function of S.

From mass conservation of the two fluids and from Darcy's law for each fluid, one can derive the following system of PDEs and algebraic relations that govern the two primary unknowns S and P:

$$\nabla \cdot \boldsymbol{v} = -(Q_n + Q_w), \tag{4.84}$$

$$\boldsymbol{v} = -\lambda_t \nabla P + \lambda_w p_c'(S)\nabla S + (\lambda_w \varrho_w + \lambda_n \varrho_n) g\boldsymbol{k}, \tag{4.85}$$

$$\phi \frac{\partial S}{\partial t} + f_w'(S)\boldsymbol{v}\cdot\nabla S = \nabla \cdot (h_w(S) p_c'(S)\nabla S) + g\frac{\partial G_w}{\partial z} + f_w(Q_n + Q_w) - Q_w, \tag{4.86}$$

$$Q_w = \frac{q_w}{\varrho_w}, \tag{4.87}$$

$$Q_n = \frac{q_n}{\varrho_n}, \tag{4.88}$$

$$\lambda_w(S) = \frac{K}{\mu_w} k_{rw}(S), \tag{4.89}$$

$$\lambda_n(S) = \frac{K}{\mu_n} k_{rn}(S), \tag{4.90}$$

$$\lambda_t(S) = \lambda_w(S) + \lambda_n(S), \tag{4.91}$$

$$k_{rw}(S) = K_{wm}\left[\frac{S - S_{wr}}{1 - S_{nr} - S_{wr}}\right]^a, \tag{4.92}$$

$$k_{rn}(S) = K_{nm}\left[\frac{1 - S - S_{nr}}{1 - S_{nr} - S_{wr}}\right]^b, \tag{4.93}$$

$$f_w(S) = \frac{\lambda_w}{\lambda_t}, \tag{4.94}$$

$$G_w(S) = h_w(S)(\varrho_n - \varrho_w), \tag{4.95}$$

$$h_w(S) = -\lambda_n(S) f_w(S). \tag{4.96}$$

The permeability of the porous medium is K (usually a tensor, but here taken as a scalar for simplicity); μ_w and μ_n are the dynamic viscosities of the wetting and non-wetting fluid, respectively; ϱ_w and ϱ_n are the densities of the wetting and non-wetting fluid, respectively; q_w and q_n are the injection rates of the wetting and non-wetting fluid through wells, respectively; S_{wr} is the irreducible saturation of the wetting fluid (i.e., $S \geq S_{wr}$); S_{nr} is the corresponding irreducible saturation of the non-wetting fluid (i.e., $(1-S) \geq S_{nr}$), K_{wn} and K_{nr} are the maximum values of the relative permeabilities

k_{rw} and k_{rn}, respectively, and a and b are given (Corey) exponents in the expressions for the relative permeabilities.

The two PDEs are of elliptic and hyperbolic/parabolic nature: (4.84) is elliptic since it is the divergence of a vector field, while (4.86) is parabolic ($h_w \geq 0$ because $p_c'(S) \geq 0$ and λ_n as well as f_w are positive since $k_{rn} > 0$ and $k_{rw} > 0$). Very often, p_c' is small so (4.86) is of hyperbolic nature, and S features very steep gradients that become shocks in the limit $p_c' \to 0$ and (4.86) is purely hyperbolic. A popular solution technique is based on operator splitting at each time level in a numerical scheme: (4.84) is solved with respect to P, given S, and (4.86) is solved with respect to S, given P.

The saturation S is a non-dimensional quantity, and so are ϕ, k_{rw}, k_{rn}, K_{wm}, K_{nm}, f_w, and f_w'. The quantity $\boldsymbol{v}$ is the total filtration velocity, i.e., the sum of the velocities of the wetting and non-wetting fluid. An associated velocity scale v_c is convenient to define. It is also convenient to introduce dimensionless fractions of wetting and non-wetting fluid properties:

$$
\begin{aligned}
\varrho &\equiv \varrho_w, \\
\varrho_n &= \varrho\alpha, \quad \alpha = \frac{\varrho_n}{\varrho_w}, \\
\mu &\equiv \mu_w, \\
\mu_n &= \mu\beta, \quad \beta = \frac{\mu_n}{\mu_w}, \\
Q &\equiv Q_w = \frac{q_w}{\varrho}, \\
Q_n &= Q\frac{\gamma}{\alpha}, \quad \gamma = \frac{q_n}{q_w}\,.
\end{aligned}
$$

We will benefit from making λ_w, λ_n, and λ_t dimensionless:

$$
\begin{aligned}
\lambda_w(S) &= \frac{K}{\mu}k_{rw}(S) = \lambda_c\bar{\lambda}_w, \quad \lambda_c = \frac{K}{\mu}, \quad \bar{\lambda}_w = k_{rw}, \\
\lambda_n(S) &= \frac{K}{\mu}\beta^{-1}k_{rn}(S) = \lambda_c\beta^{-1}\bar{\lambda}_n, \quad \bar{\lambda}_n = k_{rn}, \\
\lambda_t(S) &= \lambda_w(S) + \lambda_n(S) = \lambda_c\bar{\lambda}_t, \quad \bar{\lambda}_t = \bar{\lambda}_w + \beta^{-1}\bar{\lambda}_n\,.
\end{aligned}
$$

As we see, λ_c is the characteristic size of any “lambda” quantity, and a bar indicates as always a dimensionless variable. The above formulas imply

$$
h_w(S) = -\lambda_c\beta^{-1}\bar{\lambda}_n(S)f_w(S), \quad G_w(S) = h_w(S)\varrho(\alpha - 1)\,.
$$

Furthermore, we introduce dimensionless quantities by

$$
\bar{\boldsymbol{x}} = \frac{\boldsymbol{x}}{L}, \quad \bar{\boldsymbol{v}} = \frac{\boldsymbol{v}}{v_c}, \quad \bar{P} = \frac{P}{P_c}, \quad \bar{p}_c = \frac{p_c}{P_c}\,.
$$

Inserting the above scaled quantities in the governing PDEs results in

$$\bar{\nabla}\cdot\bar{\boldsymbol{v}} = -\frac{LQ}{v_c}(1+\alpha^{-1}\gamma), \tag{4.97}$$

$$\bar{\boldsymbol{v}} = -\frac{P_c\lambda_c}{v_cL}\bar{\lambda}_t\bar{\nabla}\bar{P} + \frac{\lambda_cP_c}{v_cL}\bar{\lambda}_w\bar{p}_c'(S)\bar{\nabla}S + \frac{g\lambda_c\varrho}{v_c}(\bar{\lambda}_w+\alpha\beta^{-1}\bar{\lambda}_n)\boldsymbol{k}, \tag{4.98}$$

$$\phi\frac{\partial S}{\partial\bar{t}} + \frac{t_cv_c}{L}f_w'(S)\bar{\boldsymbol{v}}\cdot\bar{\nabla}S = \frac{t_cP_c\lambda_c}{L^2}\bar{\nabla}\cdot(-\beta^{-1}\bar{\lambda}_n(S)f_w(S)\bar{p}_c'(S)\bar{\nabla}S) + \frac{t_cg}{L}\frac{\partial G_w}{\partial\bar{z}} + t_cf_wQ(1+\alpha^{-1}\gamma) - t_cQ\,. \tag{4.99}$$

As usual, L is taken as the characteristic length of the spatial domain. Since v_c is a velocity scale, a natural time scale is the time it takes to transport a signal with velocity v_c through the domain: $t_c = L/v_c$. The diffusion term in the equation (4.102) then gets a dimensionless fraction

$$\frac{LP_c\lambda_c}{v_cL^2}\,.$$

Forcing this fraction to be unity gives

$$v_c = \lambda_c\frac{P_c}{L}\,.$$

We realize that this is indeed a natural velocity scale if the velocity is given by the pressure term in Darcy's law. This term is K/μ times the pressure gradient:

$$\frac{K}{\mu}|\nabla P| \sim \frac{K}{\mu}\frac{P_c}{L} = \lambda_c\frac{P_c}{L} = v_c\,.$$

We have here dropped the impact of the relative permeabilities $\bar{\lambda}_w$ or $\bar{\lambda}_n$ since these are quantities that are less than or equal to unity.

The other term in Darcy's law is the gravity term that goes like $\lambda_c\varrho g$ (again dropping relative permeabilities). The ratio of the gravity term and the pressure gradient term in Darcy's law is an interesting dimensionless number:

$$\delta = \frac{\lambda_c\varrho g}{\lambda_cP_c/L} = \frac{L\varrho g}{P_c}\,.$$

This number naturally arises when we discuss the term

$$\frac{t_cg}{L}\frac{\partial G_w}{\partial\bar{z}} = -(\alpha-1)\beta^{-1}\delta(\bar{\lambda}_n'(S)f_w(S)+\bar{\lambda}_n(S)f_w'(S))\frac{\partial S}{\partial\bar{z}}$$

Introducing another dimensionless variable,

$$\epsilon = t_c Q = \frac{L^2 Q}{\lambda_c P_c},$$

we can write (4.97)-(4.99) in the final dimensionless form as

$$\bar{\nabla} \cdot \bar{\boldsymbol{v}} = -\epsilon(1 + \alpha^{-1}\gamma), \tag{4.100}$$

$$\bar{\boldsymbol{v}} = -\bar{\lambda}_t \bar{\nabla} \bar{P} + \bar{\lambda}_w \bar{p}_c'(S) \bar{\nabla} S + \delta(\bar{\lambda}_w + \alpha\beta^{-1}\bar{\lambda}_n)\boldsymbol{k}, \tag{4.101}$$

$$\begin{aligned}\phi \frac{\partial S}{\partial \bar{t}} + f_w'(S)\bar{\boldsymbol{v}} \cdot \bar{\nabla} S = -\bar{\nabla} \cdot (-\beta^{-1}\bar{\lambda}_n(S) f_w(S) \bar{p}_c'(S) \bar{\nabla} S) - \\ (\alpha - 1)\beta^{-1}\delta(\bar{\lambda}_n'(S) f_w(S) + \bar{\lambda}_n(S) f_w'(S)) \frac{\partial S}{\partial \bar{z}} + \\ \epsilon f_w(1 + \alpha^{-1}\gamma) - \epsilon\,. \end{aligned} \tag{4.102}$$

The eight input parameters L, q_w, q_n, μ_w, μ_n, ϱ_w, ϱ_n, and K are reduced to five dimensionless parameters α, β, γ, δ, and ϵ. There are six remaining dimensionless numbers to be set: K_{wm}, K_{nm}, S_{wr}, S_{nr}, a, and b.

4.7 The bidomain model in electrophysiology

The mechanical functioning of the heart is crucially dependent on correct electric signal propagation through the heart tissue. Understanding this signal propagation via mathematical modeling has therefore been a topic of increasing interest in the medical research on heart failure, stroke, and other heart-related diseases [10]. Below we list a common mathematical model, consisting of two PDEs coupled to a system of ODEs at each spatial point in the domain, and show how this model can be brought to a dimensionless form.

4.7.1 The mathematical model

A widely used mathematical model for the electric signal propagation in the heart tissue is the bidomain equations:

$$\chi C_m \frac{\partial v}{\partial t} = \nabla \cdot (\sigma_i \nabla v) + \nabla \cdot (\sigma_i \nabla u_e) - \chi I_{\mathrm{ion}} - \chi I_{\mathrm{app}}, \tag{4.103}$$

$$0 = \nabla \cdot (\sigma_i \nabla v) + \nabla \cdot ((\sigma_i + \sigma_e) \nabla u_e)\,. \tag{4.104}$$

These PDEs are posed in a spatial domain H for $t \in (0, T]$, and the symbols have the following meaning: u_e is the extracellular electric potential, v is the transmembrane potential (difference between the extracellular and intracellular potential), C_m is the capacitance of the cell membrane, χ is a membrane area to cell volume ratio, σ_i is an electric conductivity tensor for the intracellular space, and σ_e is an electric conductivity tensor for the extracellular space.

The boundary conditions are of Neumann type, but we drop these from the discussion, just to make things a bit simpler. The initial condition is typically $u_e = 0, v = v_r$, where v_r is a constant resting potential.

The PDE system is driven by $I_{\text{ion}} + I_{\text{app}}$, where I_{ion} is a reaction term describing ionic currents across the cell membrane, and I_{app} is an externally applied stimulus current. The applied current is a prescribed function, typically piecewise constant in time and space, while $I_{\text{ion}} = I_{\text{ion}}(v, s)$, where s is a state vector describing the electro-chemical state of the cells. Typical components of s are intracellular ionic concentrations and so-called gate variables that describe the permeability of the cell membrane. The dynamics of s is governed by a system of ODEs, see for instance [10] for details. The total current I_{ion} is often written as a sum of individual ionic currents:

$$I_{\text{ion}}(s,v) = \sum_{j=1}^{n} I_j(s,v), \tag{4.105}$$

where n is typically between 10 and 20 in recent models of cardiac cells. Most of the individual currents will be on the form $I_j(s,v) = g_j(s)(v - v_j)$, where v_j is the equilibrium potential of the specific ion, and $g_j(s)$ describes the membrane conductance of the particular ion channel. Without much loss of generality we can assume that this formulation is valid for all I_j, and the total ionic current can then be written in the general form

$$I_{\text{ion}}(s,v) = \sum_{j=1}^{n} I_j(s,v) = g(s)(v - v_{eq}(s)),$$

where $g(s) = \sum_{j=1}^{n} g_j(s)$ and $v_{eq}(s) = (\sum_{j=1}^{n} g_j v_j)/(\sum_{j=1}^{n} g_j)$.

As noted above, the dynamics of s is governed by an ODE system on the form

$$\frac{ds}{dt} = f(v,s)\,.$$

and the individual components of s typically have very different time scales, making any scaling of this system highly dependent on the component under study. For the present text, the focus is on tissue-level electrophysiology as described by (4.103)-(4.104), and we will proceed to scale these equations. The equations are of reaction-diffusion type, and the scaling will be based on the general non-linear reaction-diffusion equation in Section 3.2.1.

4.7.2 Scaling

Dimensionless independent variables are introduced by

$$\bar{x} = \frac{x}{L}, \quad \bar{y} = \frac{y}{L}, \quad \bar{z} = \frac{z}{L}, \quad \bar{t} = \frac{t}{t_c},$$

where L is the characteristic length scale, and t_c is the characteristic time scale. Dimensionless dependent variables are expressed as

$$\bar{v} = \frac{v - v_r}{v_p - v_r}, \quad \bar{u} = \frac{u_e}{u_c}.$$

As noted above, v_r is the resting potential, and v_p is the peak transmembrane potential. The scaling of v ensures $\bar{v} \in [0,1]$. We introduce the symbol $\Delta v = v_p - v_r$ to save space in the formulas: $\bar{v} = (v - v_r)/\Delta v$. The scale for u_e is u_c, to be determined either from simplicity of the equations or from available analysis of its magnitude.

The variable tensor coefficients σ_i and σ_e depend on the spatial coordinates and are also scaled:

$$\bar{\sigma}_i = \frac{\sigma_i}{\sigma_c}, \quad \bar{\sigma}_e = \frac{\sigma_e}{\sigma_c}.$$

For simplicity, we have chosen a common scale σ_c, but the two tensors may employ different scales, and we may also choose different scales for different directions, to reflect the anisotropic conductivity of the tissue. A typical choice of σ_c is a norm of $\sigma_i + \sigma_e$, e.g., the maximum value.

Finally, we introduce a scaling of the parameters entering the ionic current term

$$\bar{v}_{eq} = (v_{eq} - v_r)/\Delta v, \quad \bar{g} = g/g_c.$$

For the characteristic membrane conductance, g_c, a common choice is $g_c = 1/R_m$, where R_m is the membrane resistance at rest, but we will instead set $g_c = g_{\max}$, the maximum conductance of the membrane. These choices will ensure $\bar{v}_{eq}, \bar{g} \in [0,1]$.

Inserting the dimensionless variables in (4.103)-(4.104), the system of governing equations becomes

$$\begin{aligned}
\frac{\Delta v}{t_c}\chi C_m \frac{\partial \bar{v}}{\partial \bar{t}} &= \frac{\sigma_c \Delta v}{L^2}\nabla\cdot(\bar{\sigma}_i \bar{\nabla}\bar{v}) + \frac{\sigma_c u_c}{L^2}\nabla\cdot(\bar{\sigma}_i\bar{\nabla}\bar{u}) - \\
&\quad - \chi g_c \Delta v \bar{g}(s)(\bar{v} - \bar{v}_{eq}(s)) - \chi I_{\mathrm{app}}, \\
0 &= \frac{\sigma_c \Delta v}{L^2}\bar{\nabla}\cdot(\bar{\sigma}_i\bar{\nabla}\bar{v}) + \frac{\sigma_c u_c}{L^2}\nabla\cdot((\bar{\sigma}_i + \bar{\sigma}_e)\bar{\nabla}\bar{u}),
\end{aligned}$$

Multiplying the equations by appropriate factors leads to equations with dimensionless terms only:

$$
\begin{aligned}
\frac{\partial \bar{v}}{\partial \bar{t}} &= \frac{t_c \sigma_c}{\chi C_m L^2} \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \frac{t_c \sigma_c u_c}{\Delta v \chi C_m L^2} \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{u}) - \\
&\quad \frac{g_c t_c}{C_m} \bar{g}(s)(\bar{v} - \bar{v}_{eq}(s)) - \frac{t_c}{C_m \Delta v} I_{\mathrm{app}}, \\
0 &= \bar{\nabla} \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \frac{u_c}{\Delta v} \nabla \cdot ((\bar{\sigma}_i + \bar{\sigma}_e) \bar{\nabla} \bar{u}),
\end{aligned}
$$

The time scale is not so obvious to choose. As noted above, the ODE system that governs s and thereby $\bar{g}(s), \bar{v}_{eq}(s)$ may feature a wide range of temporal scales. Furthermore, even if we focus on the tissue equations and on the dynamics of v and u_e, the choice of natural time and length scales is not trivial. The equations are of reaction-diffusion nature, and the solution takes the form of a narrow wavefront of activation that propagates through the tissue. The region immediately surrounding the wavefront is characterized by large spatial and temporal gradients, while in most of the domain the variations are quite slow. The primary interest is usually on the wavefront phenomenon, so for now, we choose the time scale based on balancing the reaction and diffusion components, as described in Section 3.2.1. We consequently set the terms in front of the reaction term and the diffusion term to unity, which means

$$
\frac{t_c \sigma_c}{\chi C_m L^2} = 1, \quad \frac{t_c g_c}{C_m} = 1,
$$

and this principle determines the time and length scales as

$$
t_c = \frac{C_m}{g_c}, \quad L = \sqrt{\frac{\sigma_c}{g_c \chi}}\,.
$$

Two natural dimensionless variables then arise from the second diffusion term and the applied current term:

$$
\beta = \frac{u_c}{\Delta v}, \quad \gamma = \frac{I_{\mathrm{app}}}{g_c \Delta v}\,.
$$

In many cases it will be natural to set $u_c = \Delta v$, which of course removes the need for β, but we include the freedom to have u_c as some specified characteristic size of u_e (i.e., u_c is not a "free parameter", but is expressed by the other parameters in the problem).

The final dimensionless system becomes

$$\frac{\partial \bar{v}}{\partial \bar{t}} = \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \beta \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{u}) \tag{4.106}$$

$$- \bar{g}(s)(\bar{v} - \bar{v}_{eq}(s)) - \gamma \tag{4.107}$$

$$0 = \bar{\nabla} \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \beta \nabla \cdot ((\bar{\sigma}_i + \bar{\sigma}_e) \bar{\nabla} \bar{u})\,. \tag{4.108}$$

The two dimensionless variables in these equations have straightforward interpretations: β is the ratio of the span in the two electric potentials, and γ is ratio of the source term I_{app} and the time-derivative term of v, or the source term and the diffusion term in v.

We can insert typical parameter values to get a feel for the chosen scaling. We have

$$C_m = 1.0\,\mu\text{F cm}^{-2}, \quad g_c = g_{max} = 13.0\text{m S}\mu\text{F}^{-1} = 13.0\text{mS cm}^{-2},$$
$$\chi = 2000\text{cm}^{-1}, \quad u_c = \Delta v = 100\text{mV}, \sigma_c = 3.0\text{mS cm}^{-1}\,.$$

This gives the following values of t_c and L:

$$t_c = \frac{1.0\mu \text{ F cm}^{-2}}{13.0\mu \text{ F cm}^{-2}} = \frac{1.0}{13.0}\frac{\mu \text{ F}}{\text{mS}} \approx 0.076 \text{ ms},$$
$$L = \sqrt{\frac{\sigma_c}{\chi g_c}} = \sqrt{\frac{3.0 \text{ mS cm}^{-2}}{2000 \text{ cm}^{-1}\mu \text{ F cm}^{-}2}} \approx 0.087 \text{ mm}\,.$$

These values are both very small, which is related to our choice of $g_c = g_{\mathrm{max}}$. This choice implies that we base the scaling on the upstroke phase of the action potential, when both spatial and temporal variations are extremely high. This may therefore be a "correct" scaling exactly at the wavefront of the electrical potential, but is less relevant elsewhere. Choosing g_c to be for instance the resting conductance, which is the common choice when scaling the cable equation, may increase t_c, L by factors up to 2500 and 50, respectively. The large difference in scales reflects the difference between active, dynamic signal conduction and passive signals governed solely by electrodiffusion.

The conduction velocity is often a quantity of interest, and we could obtain an alternative relation between t_c and L by setting $CV = L/t_c$, where CV is the conduction velocity. In human cardiac tissue CV is known to be about 60 cm/s, while the choices above give

$$\frac{L}{t_c} = \frac{0.087\text{mm}}{0.076\text{ms}} \approx 144.\text{cm/s}\,.$$

Enforcing $L/t_c = 60\text{cm s}^{-1}$ gives the constraint $g_c \approx 4.8\text{mS cm}^{-2}$, and yields $L \approx 0.17$ mm and $t_c = 0.21$ ms.

4.7.3 An alternative I_{ion}

The simplest model that gives a qualitatively realistic description of the cardiac action potential is the FitzHugh-Nagumo (FHN) model. In contrast to the model (4.105) discussed above, the FHN model is completely phenomenological, with no relation to the underlying biophysics. However, the model can be parameterized to give reasonable values for the voltages, and has the advantage of giving a self-contained and relatively simple model that does not depend on externally determined variables like the s vector above. As above, we have the bidomain model given by

$$\chi C_m \frac{\partial v}{\partial t} = \nabla \cdot (\sigma_i \nabla v) + \nabla \cdot (\sigma_i \nabla u_e) - \chi I_{\text{ion}} - \chi I_{\text{app}}, \tag{4.109}$$

$$0 = \nabla \cdot (\sigma_i \nabla v) + \nabla \cdot ((\sigma_i + \sigma_e) \nabla u_e), \tag{4.110}$$

but now with

$$I_{\text{ion}} = -A[(v - v_r)(v - v_{th})(v - v_p) - w(v - v_r)(v_p - v_r)^2],$$

where w is governed by an ODE on the form

$$\frac{dw}{dt} = k(v - v_r) - lw\,.$$

Choosing $A = 4.16 \cdot 10^{-4}\text{mS}/(\text{mV}^2), v_r = -85\text{mV}, v_{th} = -68\text{mV}, v_p = 40\text{mV}, k = 4.0 \cdot 10^{-5}\text{mV}^{-1}\text{ms}^{-1}, l = 0.013\text{ms}^{-1}$, gives reasonably physiological values for v, while w is a dimensionless variable with values in $[0,1]$. The somewhat strange choice of parameter units are required because the function is cubic in v. Typical initial conditions are $v = v_r, u_e = w = 0$, and I_{app} is piecewise constant in space and time, with a typical value being $I_{app} = -50\,\text{mA}$ applied for 2 ms in certain regions of the domain.

The equations can be scaled following the same procedure as above. In addition to the dimensionless variables introduced above, we introduce

$$\alpha = \frac{v_{th} - v_r}{\Delta v},$$

and in terms of the dimensionless variables, we get

$$\begin{aligned}
\frac{\Delta v}{t_c} \chi C_m \frac{\partial \bar{v}}{\partial \bar{t}} &= \frac{\sigma_c \Delta v}{L^2} \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \frac{\sigma_c u_c}{L^2} \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{u}) - \\
&\quad - \chi A \Delta v^3 (\bar{v}(\bar{v} - \alpha)(\bar{v} - 1) - \bar{v} w) - \chi I_{\text{app}}, \\
0 &= \frac{\sigma_c \Delta v}{L^2} \bar{\nabla} \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \frac{\sigma_c u_c}{L^2} \nabla \cdot ((\bar{\sigma}_i + \bar{\sigma}_e) \bar{\nabla} \bar{u}), \\
\frac{1}{t_c} \frac{dw}{dt} &= k \Delta v \bar{v} - lw\,.
\end{aligned}$$

As above, we multiply with suitable factors to arrive at

$$\begin{aligned}
\frac{\partial \bar{v}}{\partial \bar{t}} &= \frac{\sigma_c t_c}{L^2 C_m \chi} \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \frac{\sigma_c t_c u_c}{\Delta v L^2 C_m \chi} \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{u}) - \\
&\quad - \frac{t_c A \Delta v^2}{C_m} (\bar{v}(\bar{v}-\alpha)(\bar{v}-1) - \bar{v}w) - \chi I_{\mathrm{app}}, \\
0 &= \frac{\sigma_c}{L^2} \bar{\nabla} \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \frac{\sigma_c u_c}{\Delta v L^2} \nabla \cdot ((\bar{\sigma}_i + \bar{\sigma}_e) \bar{\nabla} \bar{u}), \\
\frac{dw}{dt} &= k t_c \Delta v \bar{v} - t_c l w \,.
\end{aligned}$$

The time and length scales are again chosen by requiring balance of the reaction and diffusion term, which gives

$$t_c = \frac{C_m}{A \Delta v^2}, \quad L = \sqrt{\frac{\sigma_c}{A \Delta v^2 \chi}}, .$$

and we arrive at the final dimensionless system

$$\begin{aligned}
\frac{\partial \bar{v}}{\partial \bar{t}} &= \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \alpha \nabla \cdot (\bar{\sigma}_i \bar{\nabla} \bar{u}) - \\
&\quad - (\bar{v}(\bar{v}-\alpha)(\bar{v}-1) - \bar{v}w) - \beta, \\
0 &= \bar{\nabla} \cdot (\bar{\sigma}_i \bar{\nabla} \bar{v}) + \alpha \nabla \cdot ((\bar{\sigma}_i + \bar{\sigma}_e) \bar{\nabla} \bar{u}), \\
\frac{dw}{dt} &= \bar{k} \bar{v} - \bar{l} w,
\end{aligned}$$

where we have introduced the dimensionless numbers

$$\bar{k} = k t_c \Delta v, \quad \bar{l} = t_c l \,.$$

4.8 Exercises

Exercise 4.1: Comparison of vibration models for elastic structures

The time scale for displacement in elastic structures is, according to Section 4.1.1, $t_c = L\sqrt{\varrho/\mu}$ if we assume constant density ϱ and constant shear modulus μ for the structure. The purpose of this exercise is to compare this time scale with the time scales of related models.

a) Longitudinal waves in a bar can be modeled approximately by the PDE

$$\varrho \frac{\partial^2 u}{\partial t^2} + E \frac{\partial^2 u}{\partial x^2} = 0,$$

where $u(x,t)$ is the displacement along the bar, and E is Young's modulus, related to the shear modulus μ through

$$E = 2\mu(1+\nu),$$

where $\nu \in (0, 0.5]$ is Poisson's ratio. Find the time scale for the longitudinal waves and compare with the t_c for displacements in a three-dimensional body.

Solution. Introducing dimensionless dependent and independent variables the usual way gives us

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} + \frac{t_c^2 E}{\varrho L^2} \frac{\partial^2 \bar{u}}{\partial \bar{x}^2} = 0,$$

where L is the length scale, typically the length of the bar. The natural choice of t_c is to make the coefficient unity,

$$t_c = L\sqrt{\frac{\varrho}{E}} = L\sqrt{\frac{\varrho}{\mu}} \frac{1}{\sqrt{2(1+\nu)}} \approx 0.6L\sqrt{\frac{\varrho}{\mu}},$$

if we take $\nu = 0.3$ as a typical value.

b) Vertical vibrations of a beam are governed by the PDE

$$\rho \frac{\partial^2 u}{\partial t^2} + EI \frac{\partial^4 u}{\partial x^4} = 0,$$

where $u(x,t)$ is the vertical displacement along the beam, ρ is the mass per length of the beam, E is Young's modulus, and I is the moment of inertia. For a rectangular cross section of width b and height h, $I = \frac{1}{12}bh^3$. Compare the time scale for these vibrations with the time scale t_c for three-dimensional elasticity.

Solution. The dimensionless equation becomes

$$\frac{\partial^2 \bar{u}}{\partial \bar{t}^2} + \frac{t_c^2 EI}{\rho L^4} \frac{\partial^4 \bar{u}}{\partial \bar{x}^4} = 0 .$$

The natural choice of t_c is

$$t_c = L^2\sqrt{\frac{\rho}{EI}} = L\sqrt{\frac{\varrho}{\mu}} L\sqrt{\frac{12}{2h^2(1+\nu)}} \approx 2\frac{L^2}{h}\sqrt{\frac{\varrho}{\mu}},$$

where we have set $\nu = 0.3$ and used that $\rho = \varrho bh$ for a rectangular cross section. For a beam, $L \gg h$, so the time scale for vertical vibrations of a beam is much larger than the time scale for elastic waves in a three-dimensional body.

Exercise 4.2: A model for quasi-static poro-elasticity

Flow through a porous elastic medium may induce stress and deformation. This process is known as poro-elasticity and is governed by the following equations for a homogeneous medium:

$$(\lambda+\mu)\nabla(\nabla\cdot\boldsymbol{u})+\mu\nabla^2\boldsymbol{u} = -\alpha\nabla p-\varrho\boldsymbol{f}, \tag{4.111}$$

$$S\frac{\partial p}{\partial t} = \frac{K}{\mu_f}\nabla^2 p+\alpha\frac{\partial}{\partial t}\nabla\cdot\boldsymbol{u}, \tag{4.112}$$

where $\boldsymbol{u}(\boldsymbol{x},t)$ is the displacement field, λ and μ are Lame's elasticity parameters, $\alpha\in[0,1]$, $\boldsymbol{f}$ is the body force, here assumed constant (usually gravity, $\boldsymbol{f}=-g\boldsymbol{k}$, S is a so-called storage coefficient, $p(\boldsymbol{x},t)$ is the fluid pressure, K is the medium's permeability, μ_f is the dynamic viscosity of the fluid, and ϱ is the density of the fluid-solid mixture:

$$\varrho = (1-\phi)\varrho_s+\phi\varrho_f,$$

with ϱ_f being the density of the fluid, ϱ_s the density of the solid, and ϕ the porosity of the elastic medium. The equations are known as Biot's equations of poro-elasticity and written here in a quasi-static form where elastic waves are neglected.

Scale this partial differential equation model, assuming that λ, μ, α, $\boldsymbol{f}$, ϱ, ϕ, ϱ_s, ϱ_f, S, μ_f, and K are all constants.

Hint. The model is very similar to the equations of thermo-elasticity in Section 4.1.5.
Filename: `poroelasticity`.

Problem 4.3: Starting Couette flow

A fluid is confined in a channel with two planar walls $z=0$ and $z=H$. The fluid is at rest. At time $t=0$ the upper wall is suddenly set in motion with a velocity $U\boldsymbol{i}$. We assume that the velocity is directed along the x axis: $\boldsymbol{u}=u(x,z,t)\boldsymbol{i}$. From the equation of continuity, $\nabla\cdot\boldsymbol{u}=0$, we get that $\partial u/\partial x=0$ such that $\boldsymbol{u}=u(z,t)\boldsymbol{i}$. The boundary conditions are $\boldsymbol{u}=0$ at the lower wall $z=0$ and $\boldsymbol{u}=U\boldsymbol{i}$ at the upper wall $z=H$. Assume that the pressure is constant everywhere and that there are no body forces.

a) Start with the incompressible Navier-Stokes equations and the assumption $\boldsymbol{u}=u(z,t)\boldsymbol{i}$. Derive an initial-boundary value problem for $u(z,t)$. Scale the problem.

Solution. Inserting the simplified velocity in the original Navier-Stokes equations makes the convection term $\boldsymbol{u}\cdot\nabla\boldsymbol{u}$ vanish and ∇p vanishes since p is assumed constant (only the upper wall drives the flow). The result becomes

$$\varrho\frac{\partial u}{\partial t} = \mu\frac{\partial^2 u}{\partial z},$$

or using $\mu/\varrho = \nu$,

$$\frac{\partial u}{\partial t} = \nu\frac{\partial^2 u}{\partial z},$$

with $u(z,0) = 0$, $u(0,t) = 0$ and $u(H,t) = U$. This is a standard diffusion problem. The natural length scale is H, so $\bar{z} = z/H$. Using the well-established time scale $t_c = H^2/\nu$ and the velocity scale $u_c = U$, we get the dimensionless problem

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{\partial^2 \bar{u}}{\partial \bar{z}},$$

with $\bar{u}(\bar{z},0) = 0$, $u(0,\bar{t}) = 0$, $u(1,t) = 1$. There are no physical parameters. Having computed $\bar{u}(\bar{z},\bar{t})$, the physical solution can be retrieved as

$$u(z,t) = U\bar{u}(\bar{z}H, \bar{t}H^2/\nu)\,.$$

b) Start with the dimensionless Navier-Stokes equations and use the assumption $\bar{\boldsymbol{u}} = \bar{u}(\bar{z},\bar{t})\boldsymbol{i}$ to reduce the problem. The resulting equation now contains a Reynolds number, i.e., one more physical parameter than in a). Why is this an inferior approach to scaling the problem?

Solution. Inserting the simplified velocity in the scaled Navier-Stokes equations leads to

$$\frac{\partial \bar{u}}{\partial \bar{t}} = \frac{1}{\mathrm{Re}}\frac{\partial^2 \bar{u}}{\partial \bar{z}}\,.$$

Here we have the Reynolds number as parameter.

The major difference is the scaling used in time: $t_c = H/U$ (length scale is H here) versus $t_c = H^2/\nu$. The latter is much more suitable as it is based on a diffusion problem and the present problem is indeed a diffusion problem. Normally, ν is very small, so t_c based on diffusion is usually much larger than H/U. With an inappropriate time scale, $\partial\bar{u}/\partial\bar{t}$ is not of unit size, and we need a dimensionless number on the right-hand side to adjust the spatial derivative term to a non-unity size. The wrong scaling thereby introduces an extra (unnecessary) parameter.

c) Can you construct a heat conduction problem that has the same solution $\bar{u}(\bar{z},\bar{t})$ as in a)?

Solution. Consider a long rod with length H aligned with the z axis. The rod is isolated on the curved circular surface and kept at fixed temperatures U_0 and U_H at the ends $z=0$ and $z=H$, respectively. The initial temperature is U_0. Because of the insulated curved surface, heat can only propagate in the z direction, and a one-dimensional heat conduction equation is appropriate:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial z},$$

with $u(z,0) = U_0$, $u(0,t) = U_0$, and $u(H,t) = U_H$. We introduce a dimensionless temperature

$$\bar{u} = \frac{u - U_0}{U_H - U_0},$$

such that $\bar{u} \in [0,1]$. The standard time scale $t_c = H^2/\alpha$ is used, notifying that the length scale is H. Inserting the dimensionless variables in the governing equation results in the same problem as in a). It means that we from one solution $\bar{u}(\bar{z},\bar{t})$ can get solutions for heat conduction in rods of all lengths and materials, and with all boundary temperatures, as well as flow of any fluid between two walls with any gap and any velocity of the upper wall.

d) Describe how the scaled problem in this exercise can be solved by a program that solves the following diffusion problem with dimensions:

$$\begin{aligned}
\frac{\partial u}{\partial t} &= \alpha \frac{\partial^2 u}{\partial z} + f(x,t),\\
u(x,0) &= I(x),\\
u(0,t) &= U_0(t),\\
u(L,t) &= U_L(t).
\end{aligned}$$

Solution. Let z be named x. Set $\alpha = 1$, $f = 0$, $L = 1$, $I(x) = 0$, $U_0(t) = 0$, $U_L(t) = 1$. The resulting problem is our scaled problem from a).
Filename: `starting_Couette`.

Problem 4.4: Channel flow

We look at viscous fluid flow between two flat, infinite plates. Often, one first reduces such a problem to mathematical one-dimensional problems and then scale the model (cf. Problem 4.3), but if a general 2D/3D numerical Navier-Stokes model is to be used to solve the problem, it is more natural to just scale the full Navier-Stokes equations and then run the solver for the scaled model.

This is a problem dominated by viscous diffusion and shear stresses and where there is no convection. Argue why the standard scaling of the Navier-Stokes equations is inappropriate. Scale the equations and choose the time scale and pressure scale such that $\partial\bar{u}/\partial\bar{t}$, $\bar{\nabla}^2\bar{u}$, and $\bar{\nabla}\bar{p}$ all have unit coefficients. This naturally gives a time scale based on viscous diffusion and a pressure scale based on the shear stress $\mu U/H$, where H is the width of the channel, and U a characteristic inlet velocity. Set up the scaled problem and derive an exact solution in the stationary case.

Solution. The standard scaling of the Navier-Stokes equations applies L/U as time scale and ϱU^2 as pressure scale. Both these scales are tightly connected to convection: L/U is the time scale of convection, and ϱU^2 is the size of the convection term. In channel flow, the convection is zero, and the flow is dominated by viscous diffusion and shear stresses. It therefore makes more sense to choose a diffusion time scale and a shear stress-based pressure scale.

We introduce $\bar{x} = x/H$, $\bar{y} = y/H$, $\bar{z} = z/H$, $\bar{u} = u/U$, $\bar{p} = p/p_c$, and $\bar{t} = t/t_c$ in the equations results in the scaled Navier-Stokes equations:

$$\frac{\partial\bar{\boldsymbol{u}}}{\partial\bar{t}} + \mathrm{Re}\,\bar{\boldsymbol{u}}\cdot\bar{\nabla}\bar{\boldsymbol{u}} = -\bar{\nabla}\bar{p} + \bar{\nabla}^2\bar{\boldsymbol{u}},$$
$$\bar{\nabla}\cdot\bar{\boldsymbol{u}} = 0,$$

if we choose $t_c = H^2\varrho/\mu$ and $p_c = \mu U/H$. We see that these scales are exactly the diffusion scale and the shear scale: $\mu\partial U/\partial y \sim \mu U/H$. Re is as usual the Reynolds number, here $\varrho UH/\mu$, but it actually has no impact on the physics (as long as the flow is laminar) since the associated convective term is zero. A feature of the Navier-Stokes equations is that the solution can become unstable and result in turbulence if the Reynolds number is above a critical value. Physical experiments show that the flow is always laminar for $\mathrm{Re} < 2000$, while the critical Reynolds number for turbulence depends strongly on the roughness of the walls.

The exact solution (of the scaled problem) is derived by assuming flow along the channel: $u = (u_x(x,y,z),0,0)$, with the x axis pointing along the channel. Since $\nabla\cdot u = 0$, u cannot depend on x. The physics of channel flow is also two-dimensional so we can omit the z coordinate (more precisely: $\partial/\partial z = 0$ as nothing varies in this direction). Inserting $u = (u_x(y),0,0)$ in the (scaled) governing equations gives $u_x''(y) = \partial p/\partial x$. Differentiating this equation with respect to x shows that $\partial p/\partial x$ is a constant, here called $-\beta$. This is the driving force of the flow and specified as known in the problem.

Integrating $u_x''(y) = -\beta$ over the width of the channel, $[0,1]$, and requiring $u = 0$ at the channel walls, results in $u_x = \frac{1}{2}\beta y(1-y)$. The characteristic inlet flow in the channel, U, can be taken as the maximum inflow at $x = 1/2$, implying that $\beta = 8$. The length of the channel, L/H in the scaled model, has no impact on the result, so for simplicity we may just compute on the

unit square. The pressure can then be set to $p = 0$ at the outlet $x = 1$, giving $p(x) = 8(1-x)$ and $u_x = 4y(1-y)$.
Filename: `channel`.

Remarks. One may want to implement the Navier-Stokes equations in scaled form to have only one parameter Re in the PDEs. However, whatever we want to compute with the solver, we need to benchmark it on 2D/3D channel flow to verify that it works, and the proper scaling is different in this simple application. Therefore, the software should implement the original form of the Navier-Stokes equations, as this is the most general form of the model. In a particular application, one can derive a scaled model and set parameters in the original PDEs to mimic the scaled model. The present case of channel flow faces a problem, however, as there is no unique coefficient in front of the convective term in the original Navier-Stokes equations. We may therefore insert such a parameter in the implementation,

$$\rho \boldsymbol{u}_t + \alpha\rho\boldsymbol{u}\cdot\nabla\boldsymbol{u} = -nablap + \nu\nabla^2\boldsymbol{u} + \boldsymbol{f},$$

but the convective term will vanish for channel flow so there is little effect of Re in front of this term. Our scaling is obtained by setting $\rho = \nu = 1$ and $\alpha = \mathrm{Re}$.

References

[1] J. F. Douglas, J. M. Gasiorek, and J. A. Swaffield. *Fluid Mechanics*. Pitman, 1979.

[2] P. G. Drazin and W. H. Reid. *Hydrodynamic Stability*. Cambridge, 1982.

[3] H. P. Langtangen. *Finite Difference Computing with Exponential Decay Models*. Springer, 2016. `http://tinyurl.com/nclmcng/web`.

[4] H. P. Langtangen. *A Primer on Scientific Programming with Python*. Texts in Computational Science and Engineering. Springer, fifth edition, 2016.

[5] H. P. Langtangen and A. E. Johansen. The Parampool tutorial. `http://hplgit.github.io/parampool/doc/web/index.html`.

[6] H. P. Langtangen and A. E. Johansen. Using web frameworks for scientific applications. `http://hplgit.github.io/web4sciapps/doc/web/index.html`.

[7] H. P. Langtangen and S. Linge. *Finite Difference Computing with Partial Differential Equations*. Springer, 2016. `http://tinyurl.com/Langtangen-Linge-FDM-book`.

[8] C. C. Lin and L. A. Segel. *Mathematics Applied to Deterministic Problems in the Natural Sciences*. SIAM, 1988.

[9] J. D. Logan. *Applied Mathematics: A Contemporary Approach*. Wiley, 1987.

[10] J. Sundnes, G. T. Lines, X. Cai, B. F. Nielsen, K.-A. Mardal, and A. Tveito. *Computing the Electrical Activity in the Heart*. Monographs in Computational Science and Engineering. Springer, 2006.

[11] D. J. Tritton. *Physical Fluid Dynamics*. Van Nostrand Reinhold, 1977.

Index

angular frequency, 53
`assert`, 64

base unit, 1
Bernoulli's equation, 173
bidomain equations, 198
Biot number, 145, 184
Buckingham Pi theorem, 3

characteristic time, 19
creeping flow, 174

dimension of physical quantities, 2
dimensionless number, 27, 31, 34, 38, 117, 137, 171
dimensionless variable, 17, 19

e-folding time, 20
Eckert number, 182
Euler number, 176
exponential decay, 18

forced convection, 178
free convection, 179
frequency, 53
frequency, angular, 53
Froude number, 174

graphical web interface, 14
Grashof number, 182

`joblib`, 22, 28

length, 1
logistic equation, 38
Lotka-Volterra, competing species model, 86
Lotka-Volterra, predator-prey model, 83
low Reynolds number flow, 173

Mach number, 188
mass, 1
memoize function, 22
multiple software runs, 12

Navier-Stokes equations, 171
non-dimensionalization, 17
Nusselt number, 184

`parampool`, 9
Peclet number, 132, 178, 182
period (of oscillations), 53
phase shift, 65
`PhysicalQuantity`, 7
Pi theorem, 3

quality factor Q, 65

radians, 53

Reynolds number, 137, 171, 173, 178, 182

scaling, 17
Stokes problem, 173
Stokes' flow, 174
Strouhal number, 175

thermo-elasticity, 169
time, 1

units, 1
- British, 5
- conversion, 7
- software, 7
- US, 5

vortex shedding, 175

web interface (Parampool), 14
Weber number, 177