Notes on the Heat Equation for the Computational Differentiation Lecture

Jonathan Hüser, Uwe Naumann

October 24, 2016

Disclaimer

We use the same symbol for functions and functions evaluated at a point. We also use the same symbol for a value and it's numerical approximation. Furthermore, we don't aim for mathematical generality. If norms and scalar products are used we mean in Euclidean space unless stated otherwise. We use 0 indices like in the C programming language.

1 Finite Difference Approximations

In the following we always assume differentiability of the function f.

1.1 $f : \mathbb{R} \to \mathbb{R}$ (univariate scalar case)

The derivative can be defined as

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \ .$$

A *forward difference* approximates the derivative by fixing h to a small value like $h := |x| \cdot \sqrt{\text{eps}}$ where eps denotes machine precision:

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} \ .$$

A *backward difference* works in the same way except that it may give a different value depending on the shape of the function (see Figure 1):

$$f'(x) \approx \frac{f(x) - f(x-h)}{h} \ .$$

A *central difference* gives a better approximation but is more costly because it requires two evaluations at perturbed inputs:

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} \ .$$

1.2 $f : \mathbb{R}^n \to \mathbb{R}$ (multivariate scalar case)

The derivative is the gradient

$$\nabla f(\boldsymbol{x}) = \left(\frac{\partial f(\boldsymbol{x})}{\partial x_0} \frac{\partial f(\boldsymbol{x})}{\partial x_1} \cdots \frac{\partial f(\boldsymbol{x})}{\partial x_{n-1}} \right) \ .$$

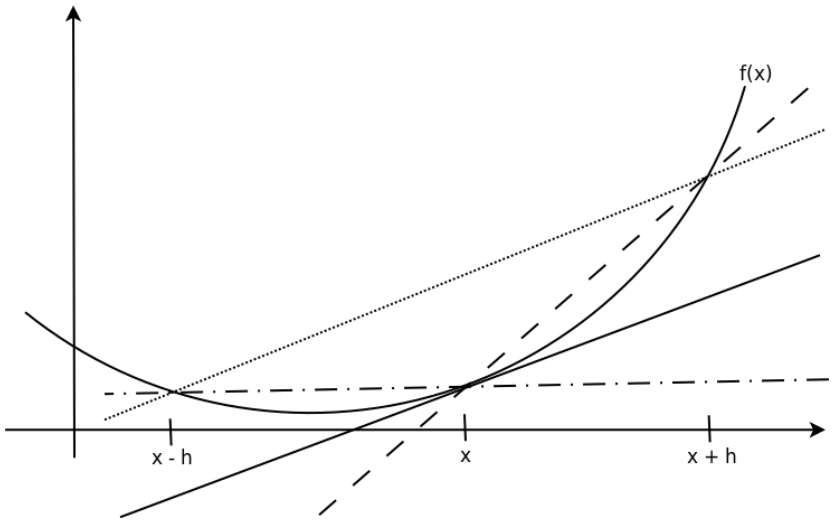

Figure 1: Forward (dashed), backward (dash-dotted) and central (dotted) finite differences. Function $f(x)$ and it's tangent at x (solid).

A finite difference approximates a gradient vector product (directional derivative):

$$\nabla f(\boldsymbol{x}) \cdot \boldsymbol{d} \approx \frac{f(\boldsymbol{x} + h \cdot \boldsymbol{d}) - f(\boldsymbol{x})}{h} .$$

We can compute all n gradient elements by approximating directional derivatives in the unit directions:

$$\frac{\partial f(\boldsymbol{x})}{\partial x_i} = \nabla f(\boldsymbol{x}) \cdot \boldsymbol{e}_i \approx \frac{f(\boldsymbol{x} + h \cdot \boldsymbol{e}_i) - f(\boldsymbol{x})}{h} \text{ for all } i \in \{0, \ldots, n-1\}$$

where $\boldsymbol{e}_i$ is the ith unit vector (has a 1 as ith entry and 0 everywhere else).

1.3 $f : \mathbb{R}^n \to \mathbb{R}^m$ (multivariate vector case)

The derivative is the Jacobian matrix

$$\nabla \boldsymbol{f}(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial f_0(\boldsymbol{x})}{\partial x_0} & \frac{\partial f_0(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_0(\boldsymbol{x})}{\partial x_{n-1}} \\ \frac{\partial f_1(\boldsymbol{x})}{\partial x_0} & \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\boldsymbol{x})}{\partial x_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{m-1}(\boldsymbol{x})}{\partial x_0} & \frac{\partial f_{m-1}(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_{m-1}(\boldsymbol{x})}{\partial x_{n-1}} \end{pmatrix} .$$

A finite difference approximates a Jacobian matrix vector product (directional derivative):

$$\nabla \boldsymbol{f}(\boldsymbol{x}) \cdot \boldsymbol{d} \approx \frac{\boldsymbol{f}(\boldsymbol{x} + h \cdot \boldsymbol{d}) - \boldsymbol{f}(\boldsymbol{x})}{h} .$$

We can compute all n columns of the Jacobian matrix by approximating directional derivatives in the unit directions:

$$\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x_i} = \begin{pmatrix} \frac{\partial f_0(\boldsymbol{x})}{\partial x_i} \\ \frac{\partial f_1(\boldsymbol{x})}{\partial x_i} \\ \vdots \\ \frac{\partial f_{m-1}(\boldsymbol{x})}{\partial x_i} \end{pmatrix} = \nabla \boldsymbol{f}(\boldsymbol{x}) \cdot \boldsymbol{e}_i \approx \frac{\boldsymbol{f}(\boldsymbol{x} + h \cdot \boldsymbol{e}_i) - \boldsymbol{f}(\boldsymbol{x})}{h} \text{ for all } i \in \{0, \ldots, n-1\}$$

where $\boldsymbol{e}_i$ is the ith unit vector (has a 1 as ith entry and 0 everywhere else).

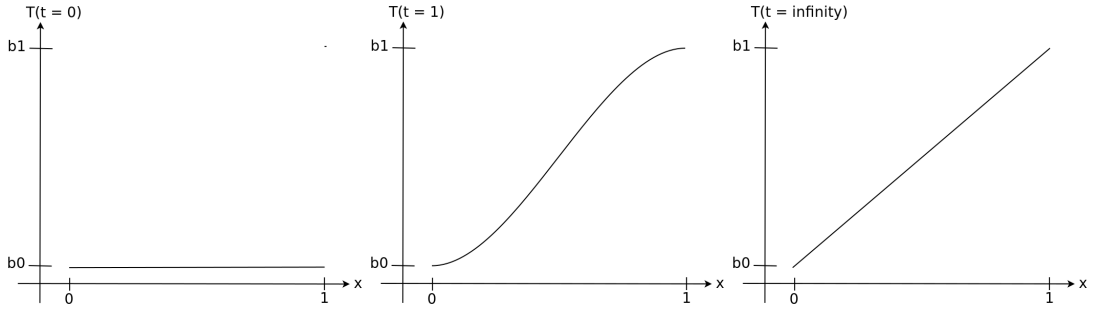

Figure 2: Temperature of the one dimensional stick at $t = 0, 1, \infty$

2 Heat Equation

2.1 Problem

We consider a model of heat spreading through a one dimensional stick that ranges from 0 to 1 in space. We model two quantities:

- Thermal diffusivity $c : [0, 1] \to \mathbb{R}^+$ that for each point on the stick describes how fast the heat spreads. Thermal diffusivity does not change over time.
- Temperature $T : [0, 1] \times \mathbb{R}^+ \to \mathbb{R}$ defined for each point on the stick and each point in time. The temperature evolution also depends on the thermal diffusivity c but we consider c as a constant for now.

The evolution of temperature in the stick over time is determined by a partial differential equation (PDE):

$$\frac{\partial T}{\partial t} = c \cdot \frac{\partial^2 T}{(\partial x)^2} \ .$$

We model the heat problem as an initial value problem where for the interior of the stick $\Omega = (0, 1)$ we have $T(t = 0) = i(x)$ for some initial temperature values $i : \Omega \to \mathbb{R}$. Furthermore, the boundaries of the stick have fixed temperature $T(x = 0) = b_0$ and $T(x = 1) = b_1$ for all points in time $t \in \mathbb{R}^+$.

We can get an intuition for the time evolution by considering the finite difference approximation for the right-hand side $\partial^2 T/(\partial x)^2$. Consider the function $f(x) = T(x, t)$ with fixed t. We approximate the second derivative via backward difference:

$$f''(x) \approx \frac{f'(x) - f'(x - h)}{h} \ .$$

We approximate both first derivatives in the above via forward differences:

$$f'(x) \approx \frac{f(x + h) - f(x)}{h} \qquad f'(x - h) \approx \frac{f(x) - f(x - h)}{h} \ .$$

Plugging in gives:

$$f''(x) \approx \frac{1}{h}\left(\frac{f(x + h) - f(x)}{h} - \frac{f(x) - f(x - h)}{h}\right) = \frac{f(x + h) - 2f(x) + f(x - h)}{h^2} \ .$$

The numerator divided by 2 contains the difference between the function evaluated at x and the average of the function evaluated in the immediate neighborhood of x:

$$\frac{1}{2}(f(x + h) + f(x - h)) - f(x).$$

The second derivative intuitively has this behavior. It described the difference between a point and it's neighborhood (in the limit). The heat equation makes this difference vanish over time and hence has a linearizing effect (see Figure 2). A linear function does not have a second derivative and at each point a linear function takes the value that is the average of it's neighborhood.

In order to solve the heat equation numerically we need to discretize the continuous variables T and c in both space and time.

2.2 Space Discretization

We discretize space into n equidistant grid points and hence get the distance between them as

$$\Delta x = \frac{1}{(n-1)}$$

We replace the right-hand side with the previously developed finite difference approximation and hence get an ordinary differential equation (ODE) that describes that dynamics at each grid point:

$$\frac{\partial T_j}{\partial t} = c \cdot \frac{\partial^2 T_j}{(\partial x)^2} \approx c \cdot \frac{T_{j+1} - 2T_j + T_{j-1}}{(\Delta x)^2} = c \cdot (n-1)^2 \cdot (T_{j+1} - 2T_j + T_{j-1}) \text{ for all } j \in \{1, \ldots, n-2\} .$$

The dynamics at the boundary points ($x = 0, 1$) are fixed to 0:

$$\frac{\partial T_j}{\partial t} = 0 \text{ for all } j \in \{0, n-1\} .$$

We write the resulting temperature ODE residual as follows:

$$\frac{\partial T}{\partial t} = r(c, \Delta x, T) .$$

The function $r(c, \Delta x, T)$ is linear in T and can be expanded into a first-order Taylor series at $T = 0$ without error term:

$$r(c, \Delta x, T) = r(c, \Delta x, 0) + \frac{\partial r}{\partial T}(c, \Delta x, 0) \cdot (T - 0) = \frac{\partial r}{\partial T}(c, \Delta x, 0) \cdot T .$$

Because $r(c, \Delta x, 0) = 0$ we can evaluate $r(c, \Delta x, T)$ as a Jacobian matrix vector product. The Jacobian matrix $\partial r(c, \Delta x, 0)/\partial T$ is only evaluated at $T = 0$ and does not depend on the time evolution. A finite difference approximation of the Jacobian matrix requires n evaluations of $r(c, \Delta, \boldsymbol{e}_i)$ with $\boldsymbol{e}_i$ the ith unit vector for all $i \in \{0, \ldots, n-1\}$ (one perturbation for each grid point). Because of the linearity of r the choice of perturbation magnitude h does not introduce any truncation error and the finite difference approximation is exact up to machine precision in this case.

2.3 Time Discretization

When solving the heat equation numerically we simulate only in the time interval $t \in (0, 1]$. We discretize time into m equidistant steps with step size

$$\Delta t = \frac{1}{m} .$$

We approximate the time derivative by a backward finite difference which results in an implicit Euler scheme for integration (simulation):

$$\frac{\partial T_j}{\partial t}(t_{k+1}) \approx \frac{T_j(t_{k+1}) - T_j(t_k)}{\Delta t} = m \cdot (T_j^{k+1} - T_j^k) .$$

We use the residual developed in the space discretization:

$$\begin{aligned} m \cdot (T^{k+1} - T^k) = r(c, \Delta x, T^{k+1}) &= \frac{\partial r}{\partial T}(c, \Delta x, 0) \cdot T^{k+1} \\ T^{k+1} - T^k &= \frac{1}{m}\frac{\partial r}{\partial T}(c, \Delta x, 0) \cdot T^{k+1} \\ -T^k &= \left(\frac{1}{m}\frac{\partial r}{\partial T}(c, \Delta x, 0) - I\right) \cdot T^{k+1} \end{aligned}$$

where I is the identity matrix.

We can solve the above time step equation as a linear system of equations $A \cdot x = b$ with

$$A = \left(\frac{1}{m}\frac{\partial r}{\partial T}(c, \Delta x, 0) - I\right)$$

and $b = -T^k$. We solve the linear system by LU decomposition. An LU factorization of A into $A = L \cdot U$ needs to only be performed once because A does not change over time. The matrices L and U are lower and upper triangular matrices so we solve $A \cdot x = b$ as $L \cdot U \cdot x = b$ in two steps: First we solve $L \cdot z = b$ as a forward substitution and then we solve $U \cdot x = z$ as a backward substitution. The matrices L and U can be stored in the same amount of memory as A because for one of them the diagonal contains only ones.

With the above time stepping procedure the simulation of the heat equation amounts to m linear system solves with identical matrix.

2.4 Parameter Calibration

Consider now the scenario where we have real world measurements (observations) $O(x)$ for the temperature at all points on the stick at time $t = 1$. To estimate the thermal diffusivitiy parameter c of our material we want to solve the least squares problem that minimizes the error between the simulation results and the observations:

$$\min_c v(c) = \int_\Omega (T(1, x, c) - O(x))^2 \mathrm{d}x \ .$$

We discretize the integral with the same number of equidistant spatial grid points as used in the simulation:

$$v(c) \approx \frac{1}{n-1} \sum_{j=0}^{n-2} (T_j^m(c) - O_j)^2 \ .$$

A gradient descent optimization of the least squares problem requires $\nabla v(c)$ the gradient of the approximated loss function in the following iterative scheme to chose improving parameterizations c_i:

$$c_{i+1} := c_i - \alpha \cdot \nabla v(c_i) \ \text{ until } \|\nabla v(c_i)\| < \varepsilon$$

where $\alpha \in \mathbb{R}^+$ is a step size and ε is a chosen termination tolerance to reach the first order optimality condition $\|\nabla v(c)\| = 0$.

The gradient $\nabla v(c)$ can be approximated using finite differences with a cost of $n+1$ evaluations of v each of which requires the full simulation of the heat equation. In the following weeks we will learn how to efficiently compute such derivative information as $\nabla v(c)$ more efficiently to enable numerical optimization.