Winfried Auzinger
Dirk Praetorius

Sommersemester 2017
10./11. Mai 2017

Übungsaufgaben zur VU Computermathematik

Serie 7

Here we mainly concentrate on the use of the data structures `Vector` and `Matrix`.

Exercise 7.1: ***Matrix representation of a linear system.***

Assume that a list is given containing m linear equations with integer coefficients in the n indexed variables `x[`k`]`, $k = 1 \ldots n$.

Example $(m = 2,\ n = 3)$:

```
[2*x[1]+x[2]-3,x[1]-x[3]+4]
```

represents the system of linear equations

$$\begin{aligned} 2\,x_1 + x_2 &= 3 \\ x_1 - x_3 &= -4 \end{aligned}$$

i.e., $A\,x = b$, with

$$A = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ -4 \end{pmatrix}$$

a) *Design a procedure which expects such a list as its argument and which returns the sequence A, b in form of a* `Matrix` *and a* `Vector`*.*

Hint: Use `? coeff` to extract the coefficients of the `x[`k`]`.

b) *Design a procedure for the inverse operation.*

Remark: For **a)** it would be somewhat tricky to detect in an automatic way what the value of n (the number of variables) is. Therefore you may specify n as an additional argument to your procedure.

Exercise 7.2: ***Matrix representation of a quadratic form.***

Assume that a quadratic form $q\colon \mathbb{R}^n \to \mathbb{R}$, i.e., a homogenous polynomial of degree 2 in the n indexed variables `x[`k`]`, $k = 1 \ldots n$, is given (with integer coefficients).

Example $(n = 3)$:

```
5*x[1]^2+4*x[1]*x[2]-x[2]*x[3]-4*x[3]^2
```

a) *Design a procedure which expects such an expression as its argument and which returns the unique* <u>*symmetric*</u> *integer $n \times n$ - matrix Q such that*[9] $q(x) = \frac{1}{2}\,x^T \cdot Q \cdot x$.

---

[9] Here, $x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ is a column vector, and x^T is the corresponding row vector.

Example ($n = 3$):

$$5\,x_1^2 + 4\,x_1\,x_2 - x_2\,x_3 - 4\,x_3^2 = \tfrac{1}{2}\,x^T \cdot Q \cdot x, \quad \text{with} \quad Q = \begin{pmatrix} 10 & 4 & 0 \\ 4 & 0 & -1 \\ 0 & -1 & -8 \end{pmatrix}$$

Hint: Use `coeff(...,2)` and `coeff(...,coeff(...))`. Again, use n as an additional argument to your procedure.

For testing examples, choose n, declare `X:=Vector(`n`,symbol=x)`, and use `LinearAlgebra[Transpose]` to convert `X` to a row vector.

b) *Design a procedure for the inverse operation.*

Exercise 7.3: ***Working with matrix functions.***

A polynomial expression $p(t) = c_0 + c_1\,t + c_2\,t^2 + \ldots$ can be applied to a quadratic matrix A, yielding $p(A) = c_0 I + c_1 A + c_2 A^2 + \ldots$ (try it). In applications (like iterative methods in linear algebra), however, application of $p(A)$ to a vector x is usually required, $y := p(A)\,x$, resulting in another vector y. This can be realized in a more efficient way.

a) *Design a procedure which expects a quadratic matrix, a polynomial function, and a vector as its arguments and which returns the vector*

$$p(A)\,x = c_0\,x + c_1\,A\,x + c_2\,A^2 x + \ldots$$

For this purpose, use an efficient Horner-like evaluation scheme:

$$p(A)\,x = c_0\,x + A\,(c_1\,x + A(\,\ldots\,))$$

Note that this involves only matrix-vector multiplications.

Hint: Organize the evaluation using a `do` loop.

Use `coeff`. `? degree(...)` returns the degree of a polynomial expression.

b) *Extend your procedure by a parameter check: A must be quadratic, and the dimensions of A and x must be compatible. Include two* `error` *exits with appropriate error messages.*

c) Let $r(x) = p(x)/q(x)$ be a rational function.

What is $r(A)$? Design a procedure which computes $r(A)\,x$.

Hint: This is only well-defined if $q(A)$ is invertible. Generate the matrix $q(A)$ by a Horner-like scheme, evaluate $p(A)\,x$ and use `LinearAlgebra[LinearSolve](`M`,`b`)`, which computes the solution y of a linear system $M\,y = b$.

Exercise 7.4: ***A special class of matrices.***

a) A quadratic matrix A is called a *Toeplitz matrix* if the values a_{jk} of its entries only depend on $j - k$. This means that the entries take constant values along each diagonal.

Example:

$$A = \begin{pmatrix} 1 & 0 & 2 & 4 \\ 3 & 1 & 0 & 2 \\ 4 & 3 & 1 & 0 \\ 0 & 4 & 3 & 1 \end{pmatrix}$$

Design a procedure `istoeplitz` *which expects a quadratic matrix as its arguments and which returns* `true` *if it is Toeplitz and* `false` *otherwise.*

b) Topelitz matrices are examples of 'data-sparse' matrices: A Toeplitz matrix is uniquely defined by its first row r and its first column c (this is still slightly redundant since $r_1 = c_1$.)

Assume that two vectors r and c of the same length (with $r_1 = c_1$) represent a Toeplitz matrix A. Design a procedure which expects r, c, and another vector x as its arguments and which computes the matrix-vector product Ax in a memory-efficient way, namely without explicitly building the matrix A.

Exercise 7.5: *Multivariate Taylor expansion.*

Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a smooth scalar function in n variables. The Taylor polynomial of degree 2 of such a function about a point $\xi \in \mathbb{R}^n$ looks as follows:

$$p_2(x;\xi) = f(\xi) + \underbrace{\nabla f(\xi)\cdot(x-\xi)}_{\text{linear form}} + \underbrace{\tfrac{1}{2}\,(x-\xi)^T\cdot(\nabla^T\nabla)f(\xi)\cdot(x-\xi)}_{\text{quadratic form}}$$

Here, x and ξ are to be interpreted as column vectors. ∇f is the *gradient* of f, i.e., the row vector consisting of the first partial derivatives of f,

$$\nabla f(\xi) = \Big(\frac{\partial f}{\partial x_1}(\xi), \;\ldots\; \frac{\partial f}{\partial x_n}(\xi)\Big)$$

$(\nabla^T\nabla)f$ is the so-called *Hessian matrix* of f; it is symmetric and contains all second partial derivatives [10] of f,

$$(\nabla^T\nabla)f(\xi) = \begin{pmatrix} \dfrac{\partial^2 f}{\partial x_1^2}(\xi) & \cdots & \dfrac{\partial^2 f}{\partial x_1\,\partial x_n}(\xi) \\ \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n\,\partial x_1}(\xi) & \cdots & \dfrac{\partial^2 f}{\partial x_n^2}(\xi) \end{pmatrix}$$

a) *Design a procedure* `ntay2` *which expects a function $f\colon \mathbb{R}^n \to \mathbb{R}$ and a vector $\xi \in \mathbb{R}^n$ as its arguments and which returns the expression $p_2(x;\xi)$ in the indexed variables* `x[1],...,x[n]` *representing x.*

Remark: It will be fine if you realize this for the case $n = 3$ only. Here it is convenient to replace the variables `x[1],x[2],x[3]` by `x,y,z`. Choose an example and compare with `? mtaylor`.

b) *Choose a function $f\colon \mathbb{R}^2 \to \mathbb{R}$ and a point $\xi \in \mathbb{R}^2$ (e.g., $\xi = (0,0)$), and use* `? plot3d` *and* `display` *to plot the graphs of the functions f and p_2 in a single 3D plot. Choose a plot range around the point ξ which is not too large (p_2 is a local approximation to f). Furthermore, verify that all partial derivatives of p_2 up to degree 2 at $x = \xi$ coincide with the corresponding derivatives of f.* (This is true by construction, according to the general principle underlying Taylor approximation.)

Exercise 7.6: *Visualization of linear mappings via parametric plots.*

An $n{\times}n$ - matrix A represents a linear mapping $x \mapsto A\,x$ from $\mathbb{R}^n$ to $\mathbb{R}^n$. We visualize the behavior of such a mapping for $n = 2$ and $n = 3$.

a) *Design a procedure expecting a numerical 2×2 - matrix A and a positive integer M as its arguments. Use polar coordinates to generate M equally spaced points (vectors) x_j on the unit circle in $\mathbb{R}^2$. Apply A to all these vectors, $y_j := A\,x_j$ for $j = 1\ldots M$. Use* `plots[pointplot]` *with option* `style=line` *and* `scaling=constrained` *to plot the resulting curve. Produce a nice plot, and also include the images of the unit vectors $x = (0,1)$ and $x = (1,0)$.*

Hint: Use `display`. How 'smooth' the resulting plot looks like will depend on A and M.

[10] More precisely: This matrix is symmetric if all second partial derivatives are continuous, because in this case, $\frac{\partial^2 f}{\partial x_j\,\partial x_k} = \frac{\partial^2 f}{\partial x_k\,\partial x_j}$ (Schwarz' Theorem).

b) *Realize a more 'elegant' version of such a procedure using a parametric version of* `? plot`*.*

Hint: First you have to understand how this works. Simplest example:

```
plot([sin(phi),cos(phi),phi=0..2*Pi],scaling=constrained)
```

generates a plot of the unit circle (this corresponds to the case $A = I$).

c) *Generalize your procedure from* **b)** *to the case $n = 3$, i.e., produce a 3D plot of the image of the unit ball in $\mathbb{R}^3$ under the mapping A.*

Hint: Use `? plot3d`. The syntax is for a parametric 3D plot is somewhat different from the 2D case. Simplest example:

```
plot3d([cos(phi)*sin(theta),sin(phi)*sin(theta),cos(theta)],
       phi=0..2*Pi,theta=0..Pi,scaling=constrained)
```

generates a 3D plot of the unit ball in $\mathbb{R}^3$.

d) $(*)$ *Extend* **c)***, adding the images of the unit vectors to the plot.*

Exercise 7.7: *Verification of a simple identity in linear algebra.*

Let A be an $m \times n$- matrix (m rows, n columns). Here we assume that $m > n$.

a) *Choose a matrix A with integer entries and full rank n (see also* `LinearAlgebra[Rank]`*). Check the identity*[11]

$$\mathbb{R}^m = \text{image}(A) \oplus \text{kernel}(A^T)$$

where the two subspaces are orthogonal to each other, $\text{image}(A) \perp \text{kernel}(A^T)$.

Here, $\text{image}(A)$ is the subspace of $\mathbb{R}^m$ spanned by the columns of A (see `LinearAlgebra[ColumnSpace]`), and $\text{kernel}(A^T)$ is the kernel of A^T (see `LinearAlgebra[NullSpace]`).

b) *Repeat* **a)** *for a matrix with rank $< n$.*

Exercise 7.8: *Your favorite package?*

Look at the help page `? index`, and select `packages.` Here you see a complete list of available packages.

Choose one of them, have a closer look, and prepare a small demo of its basic features.

There are many different packages. If you have no other special preference, you may take a closer look at the package `geometry`. *Aficionados* of combinatorics may look at `combinat` (see also `combstruct`). And there are many, many others.

[11] The proof of this identity is easy. But here we just 'verify' it by experiment.