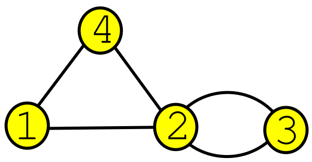
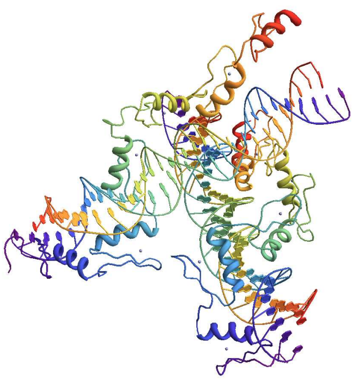


GRAPHS, NETWORKS

Linear algebra helps to understand **networks**, collection of nodes connected by edges. Networks are also called **graphs**. The **adjacency matrix** of a graph is an array of numbers defined by $A_{ij} = 1$ if there is an edge from node i to node j in the graph. Otherwise the entry is zero. An example of such a matrix appeared on an MIT blackboard in the movie "Good will hunting".

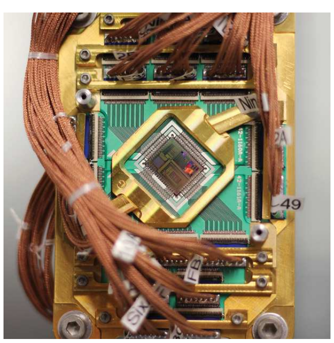
The array A helps to understand the network: assume we want to find the number of walks of length n in the network which start a vertex i and end at the vertex j. It is given by A^n_{ij}, where A^n is the n-th power of the matrix A and A^n is the n'th power of the matrix. An other application is the "page rank". The network structure of the web allows to assign a "relevance value" telling how important each node is. This is the bread and butter for a multi-billion dollar enterprise.



CHEMISTRY, MECHANICS

Complicated objects like a bridge or a protein can be modeled by finitely many parts. The bridge elements or atoms are coupled with attractive and repelling forces. The vibrations of the system are described by a differential equation $\dot{x} = Ax$, where $x(t)$ is a vector which depends on time. Differential equations are an important part of this course. Much of the theory developed to solve linear systems of equations can be used to solve differential equations.

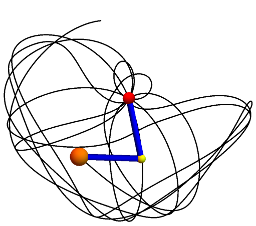
The solution $x(t) = e^{At}x(0)$ of the equation $\dot{x} = Ax$ requires to compute so called **eigenvalues** of the matrix A. These are numbers λ for which the equation $Ax = \lambda x$ has a solution. Knowing these spectral numbers is important for understanding it because the engineer can identify and damp dangerous frequencies. In chemistry or medicine, the knowledge of the vibration resonances allows to determine the shape of a molecule. Making a molecule vibrate allows to heat it. This is the principle of the microwave oven.



QUANTUM COMPUTING

A **quantum computer** is a quantum mechanical system which is used to perform computations. The state x of a machine is no more a sequence of bits like in a classical computer, but a sequence of **qubits**, where each qubit is a vector. The memory of the computer is a list of such qubits. Each computation step is a multiplication $x \mapsto Ax$ with a suitable matrix A.

Theoretically, quantum computations could speed up conventional computations significantly. They can be used for cryptological purposes. Quantum computer languages like QCL can simulate quantum computers with an arbitrary number of qubits. Whether it is possible to build quantum computers with hundreds or even thousands of qubits and performing as expected is not known. The best simulations (Oct 2018) have 56 Qbits.



CHAOS THEORY

Dynamical systems theory has the goal to predict the future of a system. At each time t, one has a map $T(t)$ on a linear space like our 3-space. The linear approximation $DT(t)$ is called the **Jacobean**. It is a matrix. If the largest eigenvalue of $DT(t)$ of T grows exponentially fast in t, then the orbit is called "chaotic".

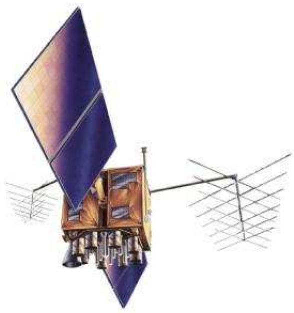
Examples of dynamical systems are a collection of stars in a galaxy, electrons in a plasma or particles in a fluid. The theoretical study is intrinsically linked to linear algebra, because stability properties often depend on linear approximations. To measure chaos, one uses linear algebra. The picture shows the double pendulum system, a chaotic system.

ERROR CORRECTION

Coding theory is used for encryption or error correction. For encryption, data vectors x are mapped into the **code** y from which it is hard to recover x. For error correction, a code is a linear subspace X of a vector space. The projection onto X corrects the error.

Linear algebra enters in different ways. Especially when designing good error correcting codes. It is a place where one can make use of the concept of image and kernel.

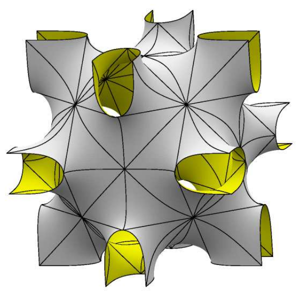
DATA COMPRESSION



Image, video and sound **compression** algorithms make use of linear transformations like the Fourier transform. One can take advantage of the fact that in the Fourier space, some information can be cut away without harm. Without data compression songs or movies would be at least 10 times larger.

Typically, a picture, a sound or a movie is cut into smaller junks. These parts are represented as vectors. If U is the Fourier transform and P is a cutoff function, then $y = PUx$ is transferred or stored on a CD, DVD or Blue-ray The receiver like the DVD player or smart phone recovers $U^T y$ which is close to x in the sense that the human eye or ear does not notice a big difference.

SOLVING EQUATIONS



When **maximizing** a function f on data which satisfy a constraint $g(x) = 0$, the method of Lagrange multipliers asks to solve a nonlinear system of equations $\nabla f(x) = \lambda \nabla g(x), g(x) = 0$ for the $(n+1)$ unknowns (x, l), where ∇f is the gradient of f.

Solving systems of nonlinear equations can be tricky. Already for systems of polynomial equations, one has to work with linear spaces of polynomials. Even if the Lagrange system is a linear system, the solution can be obtained efficiently using a solid foundation of linear algebra.

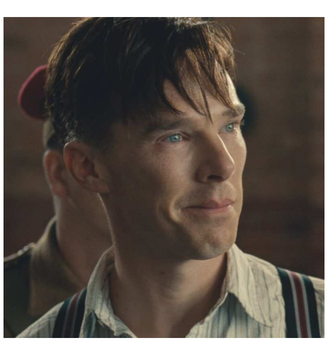
GAMES

To move around in a **computer game** requires rotations and translations to be implemented efficiently. Hardware acceleration can help to handle this. We live in a time where graphics processor power grows at a tremendous speed. Virtual reality again tries to get a foothold in the consumer market.

Rotations are represented by special matrices. For example, if an object centered at $(0,0,0)$ is turned around the y-axes by an angle ϕ, every point in the object gets transformed by the matrix

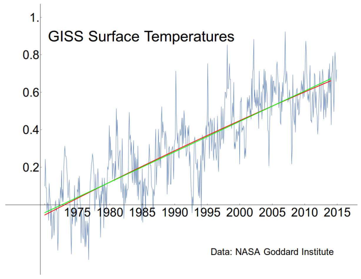
$$\begin{bmatrix} \cos(\phi) & 0 & -\sin(\phi) \\ 0 & 1 & 0 \\ \sin(\phi) & 0 & \cos(\phi) \end{bmatrix}$$

CRYPTOLOGY



Some encryptions are based on the difficulty to factor large integers n. Factoring appears to be hard. One attack is to find integers x such that x^2 has a small square y^2 reminder when dividing by n. Then $x^2 - y^2 = (x-y)(x+y)$ is divisible by n, and $x - y$ can produce a factor. One has to find many x such that x^2 has a small remainder and then use sieving methods.

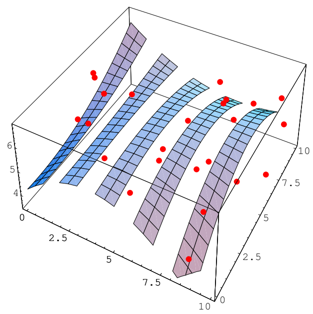
Linear algebra plays an important role in some factorization algorithms. Some use Gaussian elimination, a topic we learn early in the course. One of the factorization algorithms is the so called **quadratic sieve**. It uses linear algebra to find the factor. Still, for large integers, say with 300 digits, the best methods do not work. This difficulty allows to build encryption which would require a factorization.



BIG DATA STATISTICS

When analyzing data statistically, one is often interested in the **correlation matrix** $A_{ij} = \mathrm{E}[Y_i Y_j]$ of a random vector $X = (X_1, \dots, X_n)$ with $Y_i = X_i - E[X_i]$. This matrix can be derived from data. It sometimes even determines the random variables, if the type of the distribution is fixed.

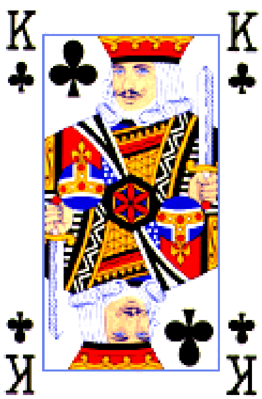
For example, if the random variables have a Gaussian (=Bell shaped) distribution, the correlation matrix together with their expectations $E[X_i]$ determines the random variables. To show this, one requires linear algebra. The magic of linear algebra is one can reduce a complex system of random variables to pairwise independent quantities. The method allows so to see what is important.



DATA FITTING

Given a bunch of data points, we would like to spot trends or use the data for predicting future outcomes. Linear algebra allows to solve this problem in a general and elegant way. For any model, which aims to represent the data points using certain type of functions, the method gives the best fit. The same idea work in higher dimensions, if we wanted to see how a certain data point depends on two data sets.

We will see this a lot, for explicit examples in this course. The simplest data fitting problem is the linear fitting which is used to find out how certain data depend on others. But knowing the method in general helps you to build your own method. A particular application is in computer vision, where from a sequence of pictures, one wants to reconstruct the object.



GAME THEORY

Abstract **games** or strategic configurations are often represented by pay-off matrices. These matrices tell the outcome when the decisions of each player are known. In the **prisoner dilemma**, each player A, B has the choice to confess or to deny. The game is described by a 2x2 matrix like $M = \begin{pmatrix} 3 & 0 \\ 5 & 1 \end{pmatrix}$.

If both prisoners A,B confess, both get 3 years. If one confesses, the other denies, the confessor gets 5, the other 0. If both deny, both only get 1 point. More generally, in a game with two players where each player can chose from n strategies, the payoff matrix is a n times n matrix M. A Nash equilibrium is a vector $p \in S = \{\sum_i p_i = 1, p_i \geq 0\}$ for which $qMp \leq pMp$ for all $q \in S$. In the above example the equilibrium is $p = (1, 0)$ when both cooperate.

NEURAL NETWORK

In **Hopfield neural network** theory the state of a system is modeled by a vector x, where each component x_j takes the values -1 or 1. If W is a symmetric $n \times n$ matrix, one can define a "learning map" $T : x \mapsto \mathrm{sign} Wx$, where the sign is taken component wise. The energy of the state is the dot product $-(x, Wx)/2$. One is interested in states of extremal energy.

For

$$W = \begin{bmatrix} 0 & 2 & 1 \\ 2 & 0 & 6 \\ 1 & 6 & 0 \end{bmatrix}$$

, we have $T([-1,-1,1]) = -1, 1, -1$ and $T([-1,1,-1]) = [1,-1,1]$. The energies are $5, 7, 7, 7$.

MARKOV PROCESSES

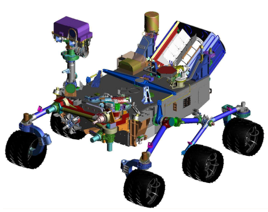
.
Suppose we have three bags containing 100 balls each. Every time, when a 5 shows up, we move a ball from bag 1 to bag 2, if the dice shows 1 or 2, we move a ball from bag 2 to bag 3, if 3 or 4 turns up, we move a ball from bag 3 to bag 1 and a ball from bag 3 to bag 2. After some time, how many balls do we expect to have in each bag?

The problem defines a **Markov chain** described by a matrix

$$\begin{bmatrix} 5/6 & 1/6 & 0 \\ 0 & 2/3 & 1/3 \\ 1/6 & 1/6 & 2/3 \end{bmatrix}$$

From this matrix, the equilibrium distribution can be read off as an eigenvector of a matrix. Eigenvectors will play an important role throughout the course.

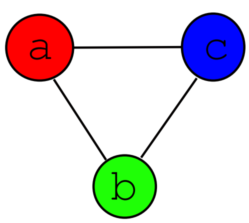
SPLINES



Computer aided design (CAD) is used for example to construct cars. One wants to interpolate points with smooth curves. One example: assume you want to find a curve connecting two points P and Q and the direction is given at each point. Find a cubic function $f(x,y) = ax^3 + bx^2y + cxy^2 + dy^3$ which interpolates.

If we write down the conditions, we will have to solve a system of 4 equations for four unknowns. Graphic artists or game designers need to have linear algebra skills also at many other topics in computer graphics.

SYMBOLIC DYNAMICS



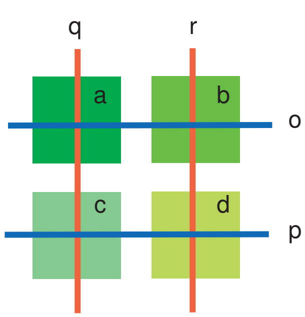
Assume that a system can be in three different states a, b, c and that transitions $a \mapsto b$, $b \mapsto a$, $b \mapsto c$, $c \mapsto c$, $c \mapsto a$ are allowed. A possible evolution of the system is then $a, b, a, b, a, c, c, c, a, b, c, a$... One calls this a description of the system with **symbolic dynamics**. This language is used in information theory or in dynamical systems theory.

The dynamics of the system is coded with a symbolic dynamical system. The transition matrix is

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}.$$

Information theoretical quantities like the "entropy" can be read off from this matrix.

TOMOGRAPHY



Tomography was studied first in 1917, and is an important tool in medical diagnosis, plasma physics or for astrophysical applications. Mathematical methods developed for the solution of this problem lead to the construction of sophisticated scanners. It is important that the inversion $h = R(f) \mapsto f$ is fast, accurate, robust and requires as few data points as possible.

A toy problem is to have 4 cells with density a, b, c, d arranged in a square. We are able and measure the light absorption by by sending light through it. Like this, we get $o = a + b$,$p = c + d$,$q = a + c$ and $r = b + d$. The problem is to recover a, b, c, d. The system of equations is equivalent to $Ax = b$, with $x = (a, b, c, d)$ and $b = (o, p, q, r)$ and

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}.$$