MVA II (HU Berlin, ERASMUS, July 3 - 12, 2008)

This *Mathematica* notebook will be displayed on the lecturer's web page.

http : // www.karlin.mff.cuni.cz/~hurt/

Mathematica Basics

References Wolfram, Hurt

Here you can look how to read *Mathematica* notebooks without *Mathematica.*

www.wolfram.com

Also a very fruitful source of information not restricted to the *Mathematica* system only and providing a variety of knowledge from assorted fields:

http : // demonstrations.wolfram.com/

General notes:

1. After typing a command, start every calculation with <Shift> + <Enter> or just <Enter> on the numeric part of the keyboard. <Enter> will just return to the next row in the input cell.

2. Built-in functions always begin with a capital. The system is case sensitive. Arguments of function are in [ ] brackets, algebraic priorities are emphasized by ( ), and { } are used for iterators ({x,0,π} or {i,1,n}) or lists (= vectors, matrices, trees, tables, &c.)

3. the cell (marked by the blue bracket on the right are basically input and output. Every input cell is introduced as In[34]:= while the corresponding output cell (after <Shift> + <Enter>) is Out[34]= .

4. If you do not need to see the output cell, it is a good idea to end the command with semicolon. The calculation is done but the output is not displayed. Appropriate for obvious results of the input or for large amounts of data in the output.

5. Put the cursor on or in the command you'd like to explain and press F1.

4 pillars of *Mathematica*

Numerics

Simple calculator:

```
In[1]:= 2 + 2
Out[1]= 4
```

π is the symbol:

$$\text{Sin}\left[\frac{\pi}{2}\right]$$

```
1
```

Numerical value of π with approx. 50 digits:

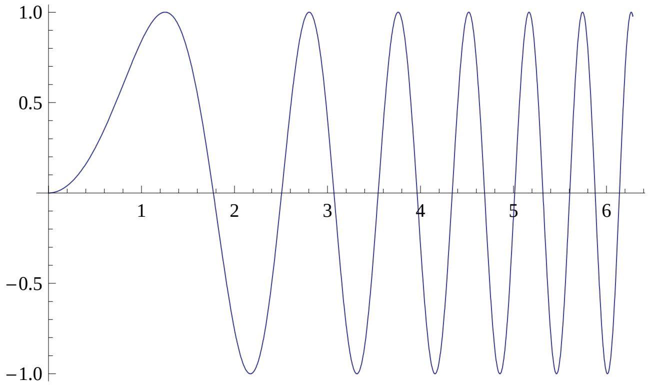
```
N[π, 50]
```

```
3.1415926535897932384626433832795028841971693993751
```

Graphics

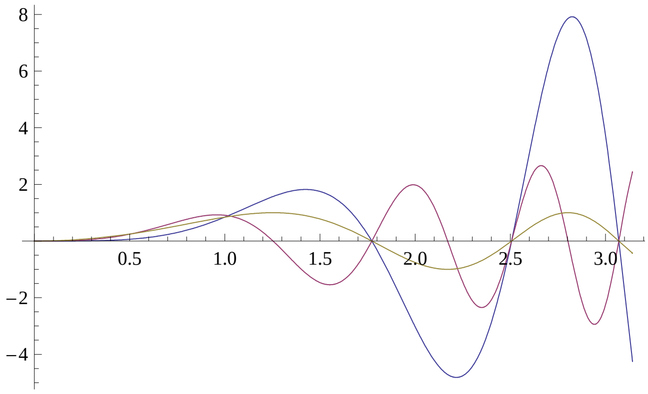
Simple:

```
Plot[ Sin[x^2], {x, 0, 2 π}]
```



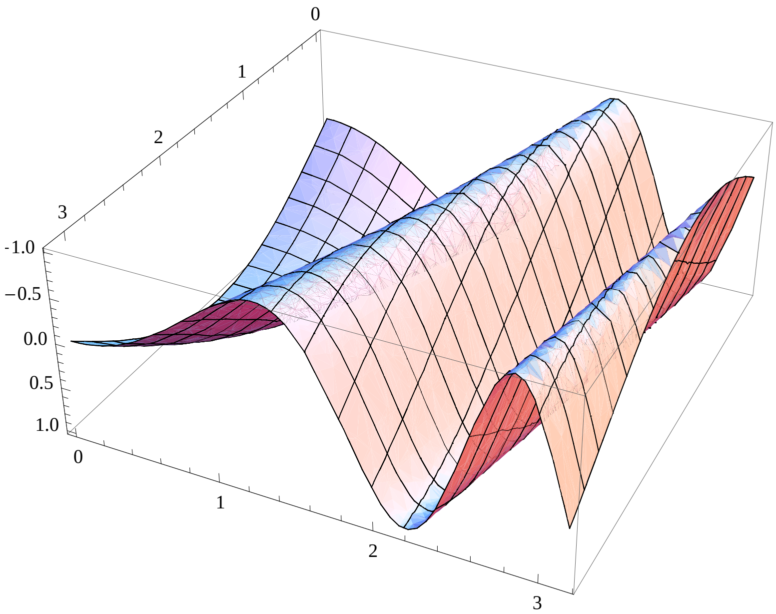
More functions in one plot :

```
Plot[{x^2 Sin[x^2], x Sin[2 x^2], Sin[x^2]}, {x, 0, π}, PlotRange → All]
```

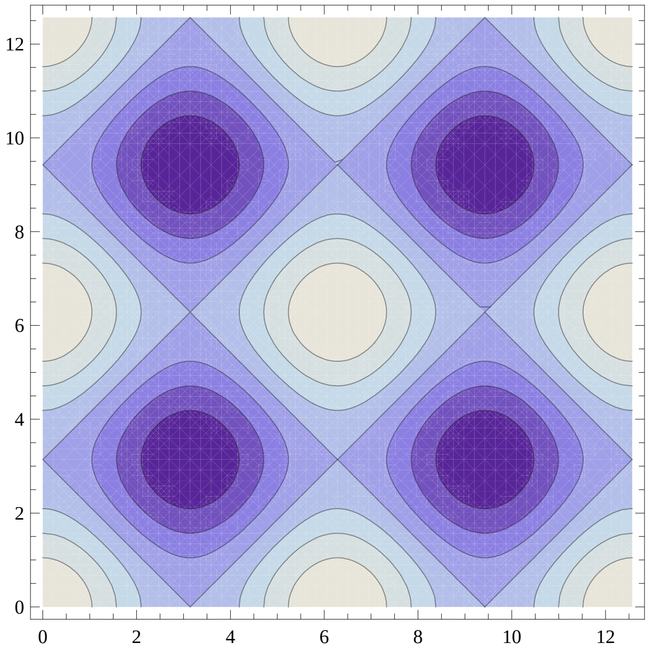


Three-dimensional graphics:

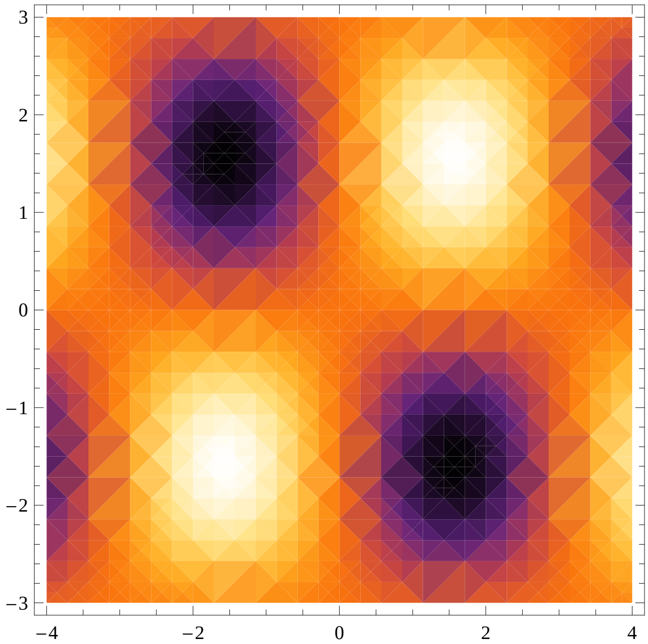
```
Plot3D[Sin[x^2 + y], {x, 0, π}, {y, 0, π}]
```




```
ContourPlot[Cos[x] + Cos[y], {x, 0, 4 Pi}, {y, 0, 4 Pi}]
```



```
DensityPlot[Sin[x] Sin[y], {x, -4, 4}, {y, -3, 3}, ColorFunction → "SunsetColors"]
```



■ Symbolics

```
e1 = (x + y)^11
```

$(x + y)^{11}$

```
e2 = Expand[e1]
```

$x^{11} + 11\,x^{10}\,y + 55\,x^{9}\,y^{2} + 165\,x^{8}\,y^{3} + 330\,x^{7}\,y^{4} + 462\,x^{6}\,y^{5} + 462\,x^{5}\,y^{6} + 330\,x^{4}\,y^{7} + 165\,x^{3}\,y^{8} + 55\,x^{2}\,y^{9} + 11\,x\,y^{10} + y^{11}$

```
Factor[e2]
```

$(x + y)^{11}$

The polynomial in the following equation may be factored, so explicit solution is achieved.

```
Solve[x^5 - 1 == 0, x]
```

$\{\{x \to 1\}, \{x \to -(-1)^{1/5}\}, \{x \to (-1)^{2/5}\}, \{x \to -(-1)^{3/5}\}, \{x \to (-1)^{4/5}\}\}$

Generally, by Solve you cannot obtain the solution if the polynomial is of order higher than 4.

```
Solve[x^5 + x^2 + 1 == 0, x]
```

```
{{x → Root[1 + #1^2 + #1^5 &, 1]}, {x → Root[1 + #1^2 + #1^5 &, 2]},
 {x → Root[1 + #1^2 + #1^5 &, 3]}, {x → Root[1 + #1^2 + #1^5 &, 4]}, {x → Root[1 + #1^2 + #1^5 &, 5]}}
```

Nevertheless, you can obtain all the roots use the numeric equivalent to Solve:

```
sol5 = NSolve[x^5 + x^2 + 1 == 0, x]
```

```
{{x → -1.19386}, {x → -0.15459 - 0.828074 i}, {x → -0.15459 + 0.828074 i},
 {x → 0.751519 - 0.784616 i}, {x → 0.751519 + 0.784616 i}}
```

For reasons that come over the scope of this lecture, we do not explain a perhaps rather strange form of the output. It is a general feature of *Mathematica*, that the results are in the form of substitutions rules. Since *Mathematica* is in some way a rewriting language, the above form of the output has the advantage that it does not change the value of any variable involved.

Nevertheless, there is a simple method to get the solution in the form of a list (roughly speaking, the symbol "/." means substitute in accordance with the substitution rules.

```
roots5 = x /. sol5
```

```
{-1.19386, -0.15459 - 0.828074 ⅈ,
 -0.15459 + 0.828074 ⅈ, 0.751519 - 0.784616 ⅈ, 0.751519 + 0.784616 ⅈ}
```

Further, we can extract the first, the second, and the last element of the above list by the following construction (we will put them into the list).

```
{First[roots5], roots5[[2]], Last[roots5]}
```

```
{-1.19386, -0.15459 - 0.828074 ⅈ, 0.751519 + 0.784616 ⅈ}
```

By Solve or NSolve we are able to solve the equations of the algebraic form. We will show it on the calculation of the internal rate of return (IRR). Suppose that $\boldsymbol{c} = \{c_0, c_1, \ldots, c_n\}$ is a cash flow in equally spaced time intervals and i is the interest rate. Then the present value is

$$\mathrm{PV}(\boldsymbol{c}, i) = \sum_{j=0}^{n} \frac{c_j}{(1+i)^j}$$

and IRR is defined as a positive solution (with respect to i) to the equation

$$\mathrm{PV}(c, i) = 0$$

First define the present value:

```
cf = {-1050, 100, 100, 100, 100, 1100};
presentvalue[i_] := cf.(1 + i)^-(Range[6]-1)
presentvalue[i]
```

$$-1050 + \frac{1100}{(1+i)^5} + \frac{100}{(1+i)^4} + \frac{100}{(1+i)^3} + \frac{100}{(1+i)^2} + \frac{100}{1+i}$$

Next we solve the equation. We get also the complex roots which have no economic meaning.

```
irr = NSolve[presentvalue[i] == 0, i]
```

```
{{i → -1.80161 + 0.582232 ⅈ}, {i → -1.80161 - 0.582232 ⅈ},
 {i → -0.69439 + 0.942471 ⅈ}, {i → -0.69439 - 0.942471 ⅈ}, {i → 0.0872374}}
```

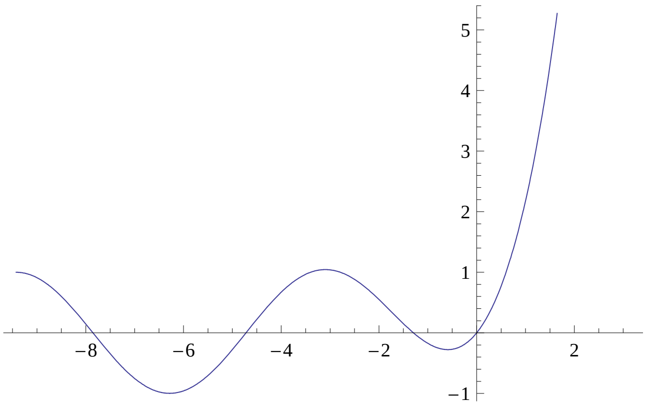
The following construction returns only the positive roots

```
Cases[i /. irr, _Real]
```

```
{0.0872374}
```

Consider the following function and try to find its roots.

```
Plot[ⅇ^x - Cos[x], {x, -3 π, π}]
```



The equation

```
eq1 = ⅇ^x - Cos[x] == 0
```

$e^x - \text{Cos}[x] == 0$

cannot be solved with Solve or NSolve. We get the warnings.

```
Solve[eq1]
NSolve[eq1]
```

Solve::tdep : The equations appear to involve the variables to be solved for in an essentially non−algebraic way. ≫

$\text{Solve}[e^x - \text{Cos}[x] == 0]$

NRoots::nnumeq :
$e^x - 1. \text{Cos}[x] == 0.$ is expected to be a polynomial equation in the variable Cos[x] with numeric coefficients. ≫

$\text{NSolve}[e^x - \text{Cos}[x] == 0]$

In this case we can use another *Mathematica* built-in function, FindRoot. In this case we must give an initial value. We must be careful since for different initial values we obtain different solutions in this specific case.

```
FindRoot[ⅇ^x - Cos[x], {x, -2}]
```

$\{x \to -1.2927\}$

```
FindRoot[ⅇ^x - Cos[x], {x, 1}]
```

$\{x \to 3.57479 \times 10^{-17}\}$

```
FindRoot[ⅇ^x - Cos[x], {x, -5}]
```

$\{x \to -4.72129\}$

Examples of integration:

```
Integrate[Cos[x], x]
```

$\text{Sin}[x]$

```
Integrate[ⅇ^(-x^2/2), {x, 0, ∞}]
```

$\sqrt{\frac{\pi}{2}}$

If the indefinite integral may be expressed in terms of known special functions, *Mathematica* does it.

```
Integrate[Sin[x^2], x]
```

$$\sqrt{\frac{\pi}{2}}\ \text{FresnelS}\left[\sqrt{\frac{2}{\pi}}\ x\right]$$

```
Integrate[Sin[x^(3/2)], {x, 0, π}]
```

$$\frac{1}{3}\,i\,\pi\left(\text{ExpIntegralE}\left[\frac{1}{3},\ -i\,\pi^{3/2}\right]-\text{ExpIntegralE}\left[\frac{1}{3},\ i\,\pi^{3/2}\right]\right)+\frac{\text{Gamma}\left[\frac{2}{3}\right]}{\sqrt{3}}$$

```
Integrate[Sin[x^3], {x, 0, π}]
```

$$\frac{1}{6}\left(i\,\pi\left(\text{ExpIntegralE}\left[\frac{2}{3},\ -i\,\pi^{3}\right]-\text{ExpIntegralE}\left[\frac{2}{3},\ i\,\pi^{3}\right]\right)+\text{Gamma}\left[\frac{1}{3}\right]\right)$$

This is the case where *Mathematica* does not know the answer and thus returns the input formula:

```
Integrate[Sin[Sin[x^2]], {x, 0, π}]
```

$$\int_0^{\pi}\text{Sin}\left[\text{Sin}\left[x^2\right]\right]\,dx$$

Still we can get the numeric answer :

```
NIntegrate[Sin[Sin[x^2]], {x, 0, π}]
```

0.694818

■ Programming

A function definition that will take a single argument (the underscore _ stands for the formal argument in the definition):

```
hilb[n_] := Table[1/(i + j - 1), {j, 1, n}, {i, 1, n}]
```

```
hilb[5] // MatrixForm
```

$$\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{pmatrix}$$

Multidimensional Scaling (MDS)

■ Remark

Since the built-in *Mathematica* functions and also the functions of AddOns packages begin always with capital letter, it is a recommended practise to define your own functions with the leading small letter. To distinguish between matrices and other quantities, I almost always use doubled symbol for a matrix, so that instead of $\boldsymbol{X}$, I write xx.

■ Demonstration that H.A.H might be not p. s. d.

Recall that $\boldsymbol{D}$ is the distance matrix. It means that $\boldsymbol{D}$ is nonnegative symmetric with zeros on the diagonal. Matrix $\boldsymbol{H}$ is the centering matrix, see the very readable definition below. The reason for $\boldsymbol{H.A.H}$ might not be positive semidivine is that $\boldsymbol{A} = -\frac{1}{2}\,\boldsymbol{D} * \boldsymbol{D}$ is <u>not</u> a matrix product but the Hadamard product (elementwise product of two matrices of the same dimensions)!!! In case of matrix product instead, matrix $\boldsymbol{B}$ must be positive semidefinite!

The following procedures are of general importance and should be carefully investigated. Function *rd* makes use of the idea that for arbitrary matrix $\boldsymbol{X}$, the matrix $\boldsymbol{X.X}^{\mathrm{T}}$ is a positive semidefinite matrix. This is very useful piece of observation in situations if we need any positive semidefinite matrix for some testing purposes. It is very difficult to construct such a matrix by hand.

```
rd[n_] := With[{xx = RandomReal[{0, 1}, {n, n}]},
  xx.Transpose[xx] - DiagonalMatrix[Diagonal[xx.Transpose[xx]]]]

centeringmatrix[n_] := IdentityMatrix[n] - 1/n Table[1, {n}, {n}]

bb[n_] := With[{dd = rd[n], hh = centeringmatrix[n]}, -1/2 hh.(dd * dd).hh]
```

Repeat the following command till you will see obviously negative eigenvalue.

```
bb[5] // Eigenvalues
```

$\{2.06609, 0.63893, 0.315146, -0.132672, 2.79182\times10^{-18}\}$

Note that since the centering matrix $\boldsymbol{H}$ is idempotent of rank $n-1$, $\boldsymbol{B}$ can not have the rank greater than this number. So at least one of the eigenvalues is zero.

■ Implementation of the method

The first argument of the input is the distance matrix and the second (optional) p is the dimension of the configuration space. If p is greater than the number of positive eigenvalues of $\boldsymbol{B}$ then the dimension is set to the number of positive eigenvalues of $\boldsymbol{B}$. If missing, then the dimension is set to the number of positive eigenvalues of matrix $\boldsymbol{B}$. We make also use of "DistanceMatrix" function from the following standard Add-On package.

```
In[80]:= Needs["HierarchicalClustering`"];
Clear[msd];
msd[dd_(* the distance matrix *), p___
  (* possibly the desired dimension of the configuration space *)] :=
 Module[{n, jj, aa, hh, bb, eigvec, eigval, eigsys, pdim, config, s1, ddhat},
  n = Length[dd];
  jj = Table[1, {n}, {n}];
  aa = -1/2 dd*dd // N;
  hh = IdentityMatrix[n] - 1/n jj;
  bb = hh.aa.hh;
  eigsys = Reverse[Sort[Transpose[Eigensystem[bb]]]];
  eigval = Map[First, eigsys];
  Print["\nNumber of points = ", n, "\nEigenvalues:\n", eigval];
  eigvec = Map[Last, eigsys];
  If[p =!= {}, pdim = p];
  pdim = Min[pdim, Count[eigval // Chop, _?Positive]];
  eigval = Take[eigval, pdim];
  eigvec = Take[eigvec, pdim];
  Print["\n\nThe configuration in ", pdim, "-dimensional space is \n",
   (config = Transpose[Sqrt[eigval] * eigvec]) // TableForm];
  ddhat = DistanceMatrix[config] // Sqrt;
  Print["Stress 1 (SS of the differences between original and estimated) = ",
   s1 = Total[Flatten[dd - ddhat]^2]];
  Print["Stress 2 (The square root of SS of the differences between original and
     estimated divided by the SS of original) = ", Sqrt[s1/Total[Flatten[dd]^2]]];
  config
 ]
```

Sets of Data

Example 1

Distances of 4 Czech cities

$$\begin{pmatrix} 0 & \text{Praha - Podebrady} & \text{Praha - Hradec} & \text{Praha - Jihlava} \\ \square & 0 & \text{Podebrady - Hradec} & \text{Podebrady - Jihlava} \\ \square & \square & 0 & \text{Hradec - Jihlava} \\ \square & \square & \square & \square \end{pmatrix}$$

In[9]:= distancematrix1 = $\begin{pmatrix} 0 & 54 & 112 & 123 \\ 54 & 0 & 58 & 102 \\ 112 & 58 & 0 & 110 \\ 123 & 102 & 110 & 0 \end{pmatrix}$;

```
In[68]:= msd[distancematrix1]
```

```
Number of points = 4
Eigenvalues:
{8305.46, 5809.07, -5.99298 × 10^-13, -0.285668}
```

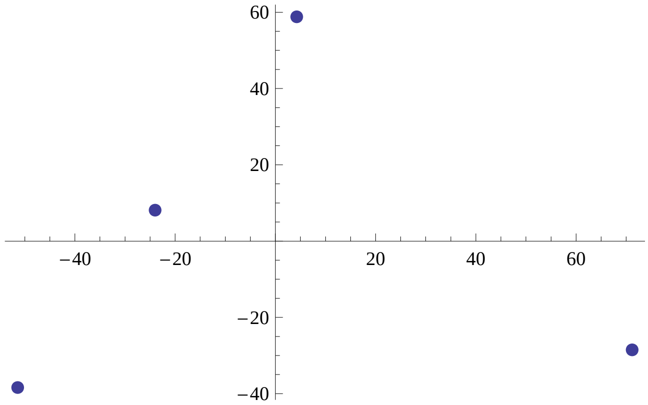
```
The configuration in 2-dimensional space is
-51.4171  -38.406
-24.0118  8.12777
4.24499   58.7832
71.1839   -28.505
Stress 1 (SS of the differences between original and estimated) = 0.0000609332

Stress 2 (The square root of SS of the differences between
   original and estimated divided by the SS of original) = 0.0000232302
```

Out[68]= `{{-51.4171, -38.406}, {-24.0118, 8.12777}, {4.24499, 58.7832}, {71.1839, -28.505}}`

Here is the plot of the resulting configuration:

In[83]:=
```
ListPlot[{{-51.417108124407065`, -38.40597969279097`},
  {-24.01183114173669`, 8.127769480186084`}, {4.244989655753748`, 58.78317163732917`},
  {71.18394961038999`, -28.504961424724318`}}, PlotStyle → PointSize[0.02]]
```

Out[83]=



▪ Example 2

Given some points in E_5 but the calculated distance matrix is based on L_1 norm (Manhattan distance). The configuration is real but the input distance matrix is not Euclidean.

In[84]:=
```
Needs["HierarchicalClustering`"]
(distancematrix2 = DistanceMatrix[{{0, 1, 9, 3, 7}, {1, 0, 8, 2, 6}, {9, 8, 0, 6, 2},
    {3, 2, 6, 0, 4}, {7, 6, 2, 4, 0}, {1, 2, 3, 4, 5}, {11, 12, 13, 14, 15}},
   DistanceFunction -> ManhattanDistance] // N) // MatrixForm
```

Out[85]//MatrixForm=

$$\begin{pmatrix} 0. & 5. & 33. & 13. & 27. & 11. & 45. \\ 5. & 0. & 32. & 10. & 26. & 10. & 48. \\ 33. & 32. & 0. & 26. & 10. & 22. & 40. \\ 13. & 10. & 26. & 0. & 20. & 10. & 50. \\ 27. & 26. & 10. & 20. & 0. & 16. & 46. \\ 11. & 10. & 22. & 10. & 16. & 0. & 50. \\ 45. & 48. & 40. & 50. & 46. & 50. & 0. \end{pmatrix}$$

Here we give the results for the desired dimension varying from 2 to 5. In *Mathematica*, it suffices to repeat the procedure with the proper number of dimensions in the form of the Table.

In[86]:= **Table[msd[distancematrix2, j], {j, 2, 5}]**

Number of points = 7
Eigenvalues:
{1890.08, 827.613, 59.7038, 23.1301, 4.19276, -1.38237×10^{-13}, -77.0031}

The configuration in 2-dimensional space is

-7.17143	12.7865
-10.1988	10.499
5.64364	-17.295
-11.6139	2.04633
-2.90328	-14.1203
-11.5266	-1.00159
37.7704	7.0851

Stress 1 (SS of the differences between original and estimated) = 150.235

Stress 2 (The square root of SS of the differences
between original and estimated divided by the SS of original) = 0.0627224

Number of points = 7
Eigenvalues:
{1890.08, 827.613, 59.7038, 23.1301, 4.19276, -1.38237×10^{-13}, -77.0031}

The configuration in 3-dimensional space is

-7.17143	12.7865	-2.32264
-10.1988	10.499	0.251483
5.64364	-17.295	0.316628
-11.6139	2.04633	6.23907
-2.90328	-14.1203	-0.995861
-11.5266	-1.00159	-3.76207
37.7704	7.0851	0.27339

Stress 1 (SS of the differences between original and estimated) = 51.5921

Stress 2 (The square root of SS of the differences
between original and estimated divided by the SS of original) = 0.036756

Number of points = 7
Eigenvalues:
{1890.08, 827.613, 59.7038, 23.1301, 4.19276, -1.38237×10^{-13}, -77.0031}

The configuration in 4-dimensional space is

-7.17143	12.7865	-2.32264	-1.05896
-10.1988	10.499	0.251483	0.807949
5.64364	-17.295	0.316628	2.43938
-11.6139	2.04633	6.23907	-0.119712
-2.90328	-14.1203	-0.995861	-3.55814
-11.5266	-1.00159	-3.76207	1.64513
37.7704	7.0851	0.27339	-0.155647

Stress 1 (SS of the differences between original and estimated) = 57.9702

```
Stress 2 (The square root of SS of the differences
   between original and estimated divided by the SS of original) = 0.0389618

Number of points = 7
Eigenvalues:
{1890.08, 827.613, 59.7038, 23.1301, 4.19276, -1.38237×10^-13, -77.0031}


The configuration in 5-dimensional space is

-7.17143  12.7865   -2.32264   -1.05896   1.03864
-10.1988  10.499    0.251483   0.807949   -1.58681
5.64364   -17.295   0.316628   2.43938    0.186286
-11.6139  2.04633   6.23907    -0.119712  0.54502
-2.90328  -14.1203  -0.995861  -3.55814   -0.427689
-11.5266  -1.00159  -3.76207   1.64513    0.282679
37.7704   7.0851    0.27339    -0.155647  -0.0381247

Stress 1 (SS of the differences between original and estimated) = 61.1384

Stress 2 (The square root of SS of the differences
   between original and estimated divided by the SS of original) = 0.0400123

Out[86]= {{{-7.17143, 12.7865}, {-10.1988, 10.499}, {5.64364, -17.295}, {-11.6139, 2.04633},
   {-2.90328, -14.1203}, {-11.5266, -1.00159}, {37.7704, 7.0851}},
  {{-7.17143, 12.7865, -2.32264}, {-10.1988, 10.499, 0.251483},
   {5.64364, -17.295, 0.316628}, {-11.6139, 2.04633, 6.23907},
   {-2.90328, -14.1203, -0.995861}, {-11.5266, -1.00159, -3.76207},
   {37.7704, 7.0851, 0.27339}}, {{-7.17143, 12.7865, -2.32264, -1.05896},
   {-10.1988, 10.499, 0.251483, 0.807949}, {5.64364, -17.295, 0.316628, 2.43938},
   {-11.6139, 2.04633, 6.23907, -0.119712}, {-2.90328, -14.1203, -0.995861, -3.55814},
   {-11.5266, -1.00159, -3.76207, 1.64513}, {37.7704, 7.0851, 0.27339, -0.155647}},
  {{-7.17143, 12.7865, -2.32264, -1.05896, 1.03864},
   {-10.1988, 10.499, 0.251483, 0.807949, -1.58681}, {5.64364, -17.295, 0.316628,
    2.43938, 0.186286}, {-11.6139, 2.04633, 6.23907, -0.119712, 0.54502},
   {-2.90328, -14.1203, -0.995861, -3.55814, -0.427689}, {-11.5266, -1.00159,
    -3.76207, 1.64513, 0.282679}, {37.7704, 7.0851, 0.27339, -0.155647, -0.0381247}}}
```

7 selected stocks (2 Frankfurt, 5 NYSE)

■ Reading and Exporting the Data to Stocks7.CSV

Real data from *Mathematica* Integrated Data Sources, till approx. July 9, 2008. If you are not connected to the Internet you can get the most actual data or play with the older data just changing "Jan. 1, 2008" to "Jan. 1, 2001" , say.   If you are not connected to the Internet just skip this Section and continue with the next Section **Importing Data from the External File Stocks7.CSV**

```
In[70]:= (* Do not calculate if you are not connected to the Internet!!! *)
priceIBM = FinancialData["IBM", "Jan. 1, 2008"];
priceGE = FinancialData["GE", "Jan. 1, 2008"];
priceBMW = FinancialData["F:BMW", "Jan. 1, 2008"];
priceGM = FinancialData["GM", "Jan. 1, 2008"];
priceJNJ = FinancialData["JNJ", "Jan. 1, 2008"];
priceVWS = FinancialData["F:VWS", "Jan. 1, 2008"];
priceABB = FinancialData["NYSE:ABB", "Jan. 1, 2008"];
```

```
In[77]:= (* Do not calculate if not connected !!! *)
prices = {priceIBM, priceGE, priceBMW, priceGM, priceJNJ, priceVWS, priceABB};
```

Exporting prices to file "Stocks7.CSV"

```
In[79]:= (* Do not calculate if not connected !!! *)
         Export["C:\\Dokumenty\\VPFPM\\Berlin2008\\Stocks7.CSV", prices]

Out[79]= C:\Dokumenty\VPFPM\Berlin2008\Stocks7.CSV
```

■ Importing Data from the External File Stocks7.CSV

```
In[87]:= im = Import["C:\\Dokumenty\\VPFPM\\Berlin2008\\Stocks7.CSV"];

         prices = {priceIBM, priceGE, priceBMW, priceGM, priceJNJ, priceVWS, priceABB} =
            Map[ToExpression, im, ∞];
```

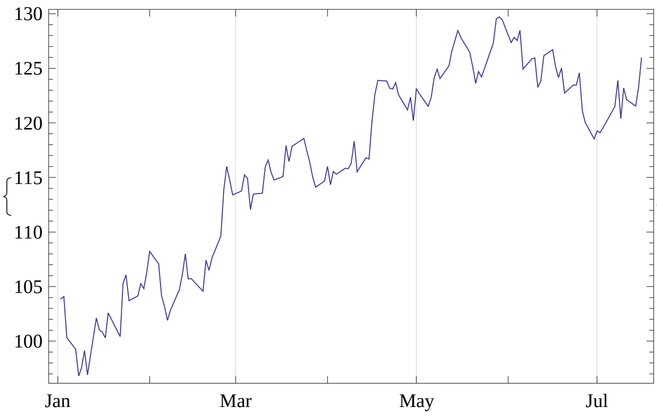
Showing how you can get parts of the lists from *Mathematica*.

```
In[92]:= prices // Last // First

Out[92]= {{2008, 1, 2}, 28.64}

In[93]:= priceABB // First

Out[93]= {{2008, 1, 2}, 28.64}

In[94]:= priceABB // First // First

Out[94]= {2008, 1, 2}

In[95]:= Map[Last, prices]

Out[95]= {{{2008, 7, 16}, 125.94}, {{2008, 7, 16}, 27.68},
          {{2008, 7, 11}, 28.89}, {{2008, 7, 16}, 11.48},
          {{2008, 7, 16}, 68.19}, {{2008, 7, 11}, 82.56}, {{2008, 7, 16}, 27.83}}
```

■ Plotting the Prices

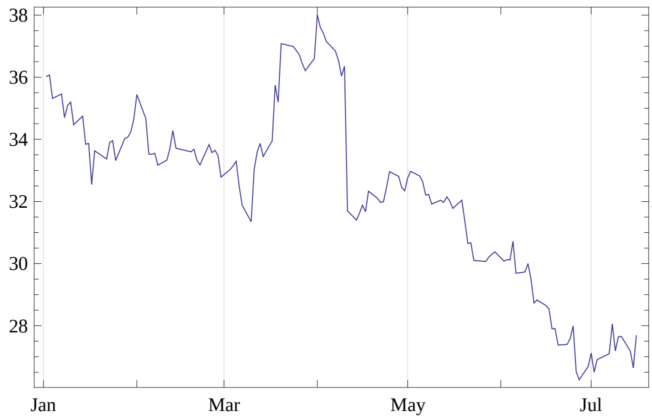
```
In[97]:= Map[DateListPlot[#, Joined → True] &, prices]
```

Out[97]= {



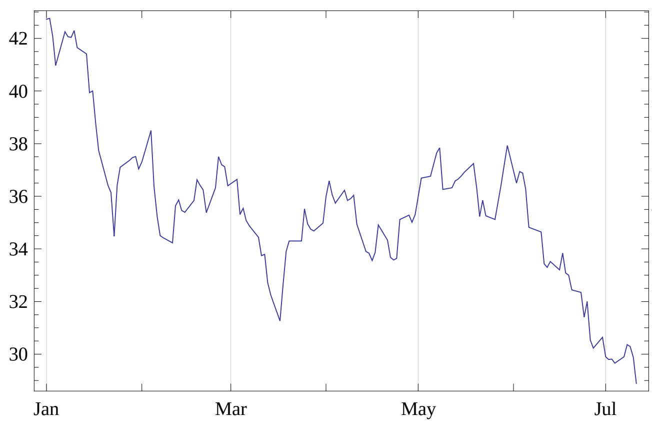
130
125
120
115
110
105
100
Jan
Mar
May
Jul

,



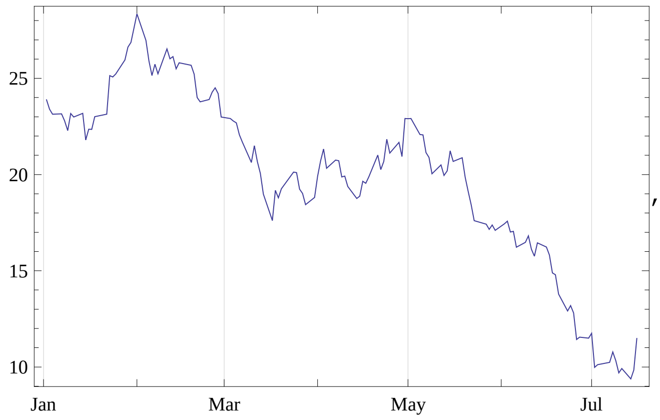
38
36
34
32
30
28
Jan
Mar
May
Jul

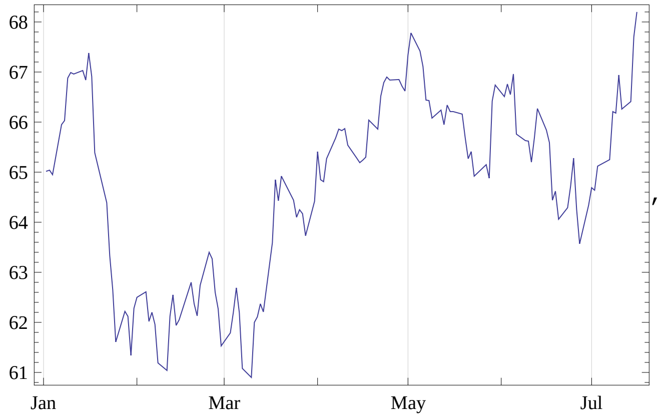
,

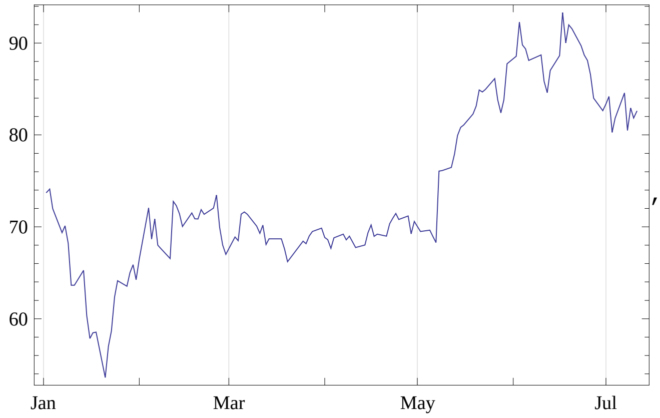


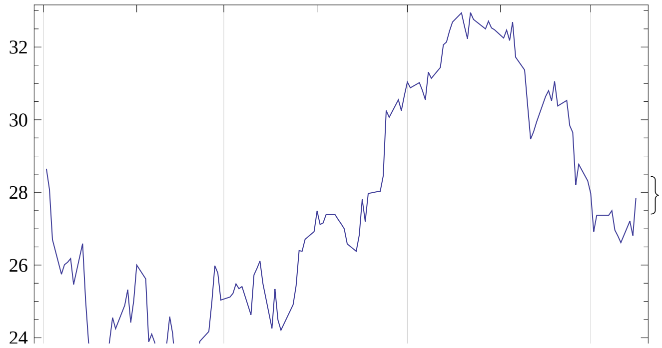
42
40
38
36
34
32
30
Jan
Mar
May
Jul

,


25
20
15
10
Jan
Mar
May
Jul
,


68
67
66
65
64
63
62
61
Jan
Mar
May
Jul
,


90
80
70
60
Jan
Mar
May
Jul
,


32
30
28
26
24
}

Find a list of NYSE stocks whose ticker symbols begin with IB :

In[99]:=
```
FinancialData["NYSE:AB*", "Lookup"]
```

Out[99]=
```
{NYSE:AB, NYSE:ABA, NYSE:ABB, NYSE:ABC, NYSE:ABD, NYSE:ABG,
 NYSE:ABI, NYSE:ABK, NYSE:ABM, NYSE:ABN, NYSE:ABN-PE, NYSE:ABN-PF,
 NYSE:ABN-PG, NYSE:ABR, NYSE:ABT, NYSE:ABW-PA, NYSE:ABX, NYSE:ABY}
```

■ Find a list of Frankfurt stocks whose ticker symbols begin with BM :

In[100]:=
```
FinancialData["F:BM*", "Lookup"]
```

Out[100]=
```
{F:BM1, F:BM2, F:BM6, F:BM8, F:BM93, F:BM9A, F:BME, F:BMI, F:BMK,
 F:BMM, F:BMO, F:BMS, F:BMT, F:BMU, F:BMW, F:BMW3, F:BMW5, F:BMX1, F:BMY}
```

■ Consolidation of the Data

We use data from two Stock Exchanges and the trading dates do not coincide. To compare the prices and/or returns we must first select the common trading dates and then extract the prices correspondingly. Check separately the parts of the following rather complex commands. You must also look how pure functions (# &) work, see Help.

In[106]:=
```
commondates = Apply[Intersection, Map[First[Transpose[#]] &, prices]];
commondates // Last
commondates // Length
```

Out[107]=
```
{2008, 7, 11}
```

Out[108]=
```
129
```

Here are two constructions how to extract the date and price of the IBM stock of April 18, 2008.

In[109]:=
```
Select[priceIBM, First[#] == {2008, 4, 18} &]
Select[priceIBM, Function[x, First[x] == {2008, 4, 18}]]
```

Out[109]=
```
{{{2008, 4, 18}, 123.89}}
```

Out[110]=
```
{{{2008, 4, 18}, 123.89}}
```

By this database construction we obtain the database containing the dates and prices that are quoted on both Exchanges on the same dates.

In[111]:=
```
consolidateddata =
   Map[Function[y, Map[Select[y, Function[x, First[x] == #]] &, commondates]], prices];
```

Number of companies:

In[116]:=
```
consolidateddata // Length
```

Out[116]=
```
7
```

A shortened output to see the structure of the data.

In[117]:=
```
Short[consolidateddata, 10]
```

Out[117]//Short=
```
{{{{{2008, 1, 2}, 103.87}}, {{{2008, 1, 3}, 104.08}},
  {{{2008, 1, 4}, 100.33}}, {{{2008, 1, 7}, 99.26}}, {{{2008, 1, 8}, 96.82}},
  {{{2008, 1, 9}, 97.54}}, <<117>>, {{{2008, 7, 3}, 119.54}}, {{{2008, 7, 7}, 121.5}},
  {{{2008, 7, 8}, 123.88}}, {{{2008, 7, 9}, 120.4}}, {{{2008, 7, 10}, 123.18}},
  {{{2008, 7, 11}, 122.12}}}, {<<1>>}, <<3>>, {<<1>>}, {<<1>>}}
```

For a further need we only need prices. Here is the method how to get the prices of the first stock, i. e. IBM:

In[118]:=
```
Map[Last, Flatten[consolidateddata // First, 1]]
```

Out[118]=
```
{103.87, 104.08, 100.33, 99.26, 96.82, 97.54, 99.13, 96.9, 102.12, 101.03, 100.83, 100.3,
 102.59, 100.42, 105.27, 106.07, 103.7, 104.15, 105.27, 104.82, 106.27, 108.22, 107.08,
 104.19, 103.17, 101.92, 102.85, 104.71, 106.1, 107.98, 105.7, 105.73, 104.57, 107.41,
 106.49, 107.63, 109.63, 113.91, 115.99, 114.77, 113.4, 113.76, 115.24, 114.92, 112.06,
 113.48, 113.55, 116.02, 116.59, 115.44, 114.76, 117.93, 116.46, 117.85, 117.49,
 116.43, 115.05, 114.1, 114.67, 116.02, 114.34, 115.55, 115.29, 115.84, 115.8, 116.29,
 118.3, 115.53, 116.8, 116.69, 119.98, 122.58, 123.89, 123.84, 123.17, 123.1, 123.68,
 122.58, 121.19, 122.35, 120.21, 122.68, 121.53, 122.32, 124.14, 124.92, 124.06,
 125.24, 126.58, 127.52, 128.46, 127.82, 126.49, 125.18, 123.62, 124.7, 124.2, 127.32,
 129.54, 129.71, 129.43, 127.36, 127.84, 127.55, 128.47, 124.94, 125.94, 123.25,
 123.85, 126.15, 126.71, 125.1, 124.16, 125.02, 122.74, 123.46, 123.46, 124.58,
 121.13, 120.05, 118.53, 119.27, 119.1, 119.54, 121.5, 123.88, 120.4, 123.18, 122.12}
```

And similarly, by a little bit more complicated construction, we get the prices of all 7 companies. The command "Dimensions" shows that from now we deal with the matrix with 7 rows representing stocks and 129 columns representing trading days.

In[119]:=
```
consolidatedprices =
  Map[Map[Last, Flatten[consolidateddata[[#]], 1]] &, Range[Length[consolidateddata]]];
Dimensions[consolidatedprices]
```

Out[120]= {7, 129}

■ The Means, Covariances, and Correlations (prices)

In[121]:=
```
Mean[consolidatedprices // Transpose]
```

Out[121]= {116.014, 32.4125, 35.5024, 20.1912, 64.688, 73.9696, 27.6047}

In[122]:=
```
Covariance[consolidatedprices // Transpose] // MatrixForm
```

Out[122]//MatrixForm=

$$\begin{pmatrix} 81.5446 & -14.2535 & -12.8073 & -25.3511 & 7.22583 & 56.7062 & 21.9976 \\ -14.2535 & 8.04738 & 4.93913 & 9.21329 & -0.950385 & -18.9487 & -3.87542 \\ -12.8073 & 4.93913 & 7.87808 & 8.34078 & 0.633695 & -10.4177 & -0.667653 \\ -25.3511 & 9.21329 & 8.34078 & 19.0448 & -3.26447 & -27.3465 & -6.17189 \\ 7.22583 & -0.950385 & 0.633695 & -3.26447 & 3.31916 & 4.1993 & 3.38244 \\ 56.7062 & -18.9487 & -10.4177 & -27.3465 & 4.1993 & 77.3012 & 18.6494 \\ 21.9976 & -3.87542 & -0.667653 & -6.17189 & 3.38244 & 18.6494 & 8.76767 \end{pmatrix}$$

In[123]:=
```
Correlation[consolidatedprices // Transpose] // MatrixForm
```

Out[123]//MatrixForm=

$$\begin{pmatrix} 1. & -0.556412 & -0.5053 & -0.643295 & 0.439214 & 0.714233 & 0.822688 \\ -0.556412 & 1. & 0.620316 & 0.744217 & -0.18389 & -0.759729 & -0.46137 \\ -0.5053 & 0.620316 & 1. & 0.680939 & 0.123924 & -0.422153 & -0.0803338 \\ -0.643295 & 0.744217 & 0.680939 & 1. & -0.410592 & -0.712723 & -0.477626 \\ 0.439214 & -0.18389 & 0.123924 & -0.410592 & 1. & 0.262162 & 0.627009 \\ 0.714233 & -0.759729 & -0.422153 & -0.712723 & 0.262162 & 1. & 0.716358 \\ 0.822688 & -0.46137 & -0.0803338 & -0.477626 & 0.627009 & 0.716358 & 1. \end{pmatrix}$$

■ Testing Random Walk Hypothesis

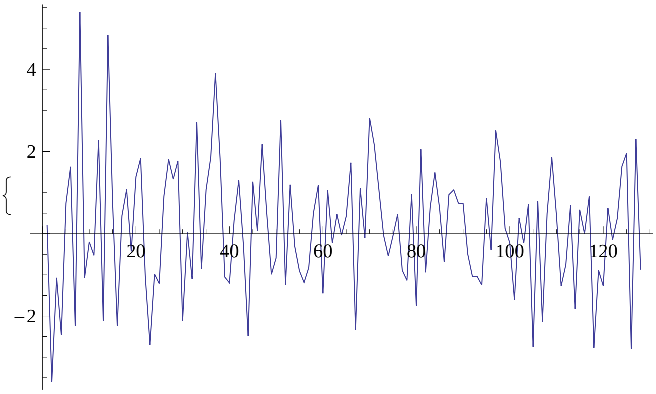
Next time.

■ Returns and their Characteristics

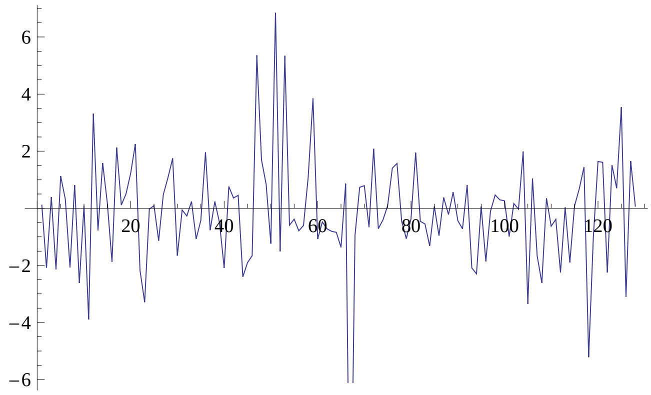
In[129]:=
```
(* Returns given in per cent *)
returns = 100 * Map[(Rest[consolidatedprices⟦#⟧] - Most[consolidatedprices⟦#⟧]) /
      Most[consolidatedprices⟦#⟧] &, Range[Length[consolidateddata]]];
```

In[130]:=
```
Map[ListPlot[#, Joined → True] &, returns]
```

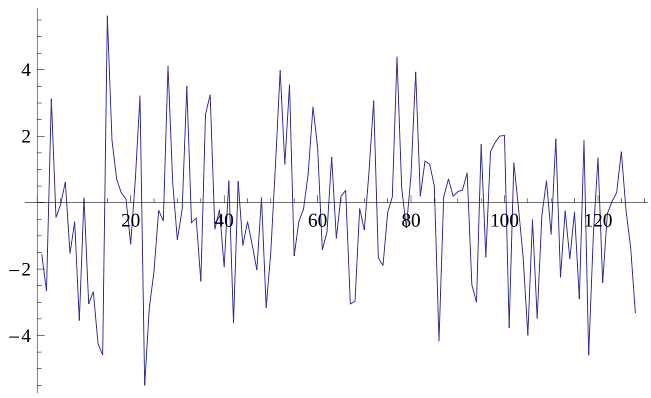
Out[130]= {



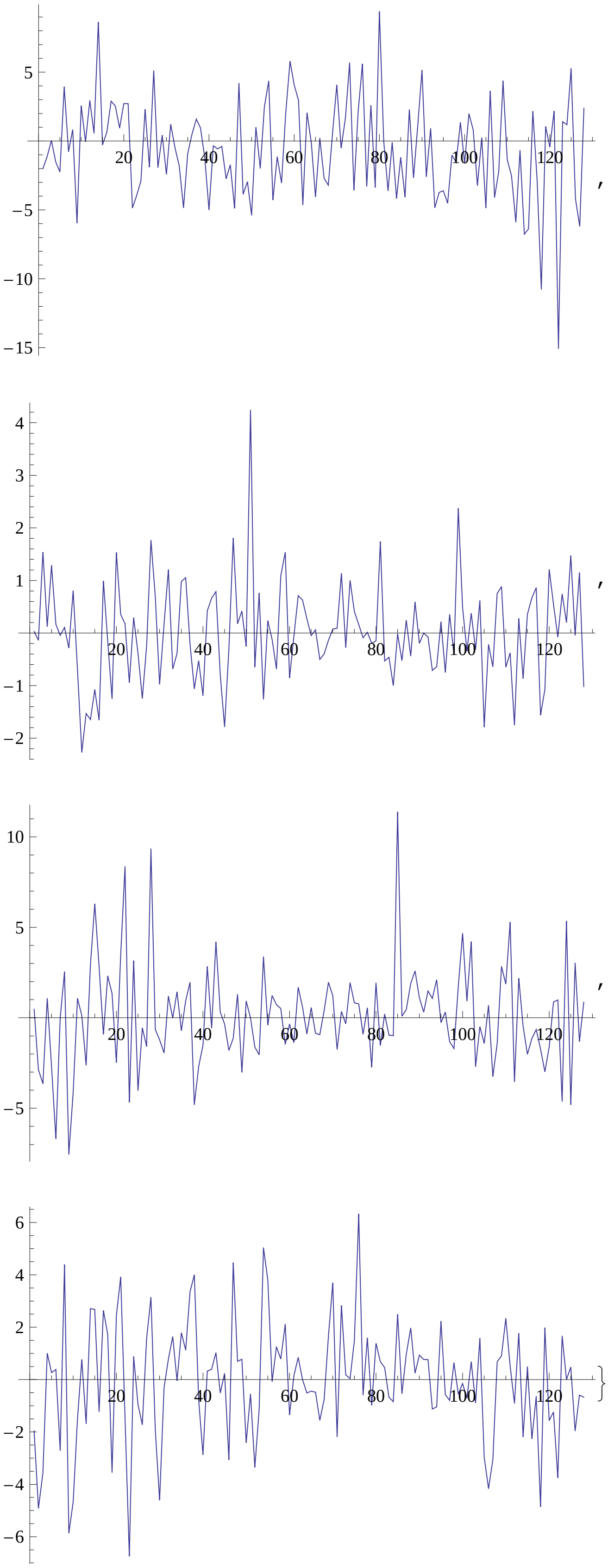
,



,



,



Expected returns estimated from the historical time series. The maximum expected returns is for IBM, position 1.

In[135]:=
```
(expectedreturns = Mean[returns // Transpose]) // TableForm
{Max[expectedreturns], Position[expectedreturns, Max[expectedreturns]]}
```

Out[135]//TableForm=
```
0.138606
-0.184389
-0.283755
-0.618576
0.0191205
0.126253
-0.0318733
```

Out[136]=
```
{0.138606, {{1}}}
```

In[137]:=
```
(covariancematrixofreturns = Covariance[returns // Transpose]) // MatrixForm
```

Out[137]//MatrixForm=

$$\begin{pmatrix} 2.44344 & 1.84727 & 0.848831 & 2.06649 & 0.453077 & 0.39021 & 1.81755 \\ 1.84727 & 4.32468 & 1.49482 & 3.46798 & 1.04532 & 0.169168 & 1.90484 \\ 0.848831 & 1.49482 & 4.44724 & 1.51603 & 0.713154 & 1.80391 & 1.45582 \\ 2.06649 & 3.46798 & 1.51603 & 12.9698 & 0.667194 & 0.394081 & 3.52243 \\ 0.453077 & 1.04532 & 0.713154 & 0.667194 & 0.882548 & 0.080174 & 0.44031 \\ 0.39021 & 0.169168 & 1.80391 & 0.394081 & 0.080174 & 7.76999 & 2.23644 \\ 1.81755 & 1.90484 & 1.45582 & 3.52243 & 0.44031 & 2.23644 & 5.06911 \end{pmatrix}$$

Now look on the expected return and risk (= standard deviation), all in per cent. We see that the displayed results are in the contradiction with the assumptions of the efficient market. The most risky is GM even with the highest negative expected return, but also other some of the assets with negative returns are more risky than those with the positive return.

Expected returns and standard deviations

In[157]:=
```
{expectedreturns, covariancematrixofreturns // Diagonal // Sqrt} // Transpose // TableForm
```

Out[157]//TableForm=

0.138606	1.56315
-0.184389	2.07959
-0.283755	2.10885
-0.618576	3.60137
0.0191205	0.93944
0.126253	2.78747
-0.0318733	2.25147

In[158]:=
```
(correlationmatrixofreturns = Correlation[returns // Transpose]) // MatrixForm
```

Out[158]//MatrixForm=

$$\begin{pmatrix} 1. & 0.568268 & 0.257499 & 0.367084 & 0.308533 & 0.0895544 & 0.516439 \\ 0.568268 & 1. & 0.340854 & 0.463054 & 0.535063 & 0.029183 & 0.406832 \\ 0.257499 & 0.340854 & 1. & 0.199616 & 0.359972 & 0.306874 & 0.306618 \\ 0.367084 & 0.463054 & 0.199616 & 1. & 0.197204 & 0.0392562 & 0.434419 \\ 0.308533 & 0.535063 & 0.359972 & 0.197204 & 1. & 0.0306164 & 0.208173 \\ 0.0895544 & 0.029183 & 0.306874 & 0.0392562 & 0.0306164 & 1. & 0.356353 \\ 0.516439 & 0.406832 & 0.306618 & 0.434419 & 0.208173 & 0.356353 & 1. \end{pmatrix}$$

Maximum correlation is between IBM and GE returns:

In[159]:=
```
With[{c = correlationmatrixofreturns},
 {Max[c - IdentityMatrix[Length[c]]], Position[c, Max[c - IdentityMatrix[Length[c]]]]}]
```

Out[159]=
```
{0.568268, {{1, 2}, {2, 1}}}
```

Eigenvalues of the covariance matrix

In[160]:=
```
covariancematrixofreturns // Eigenvalues
```
Out[160]=
```
{17.247, 8.8706, 4.60628, 3.21953, 2.24979, 1.16317, 0.550481}
```

Eigenvalues of the correlation matrix

In[161]:=
```
correlationmatrixofreturns // Eigenvalues
```
Out[161]=
```
{2.91806, 1.19401, 0.978723, 0.643586, 0.537518, 0.406046, 0.322056}
```

- ## Markowitz portfolio, short sales allowed

Lagrange function generally, μ prescribed expected return on portfolio

In[162]:=
```
Clear[lagrange];
lagrange[xx_, rr_, vv_, λ1_, λ2_, μ_] :=
 Module[{boldone},
  boldone = Table[1, {Length[xx]}];
  1/2 xx.vv.xx + λ1 (1 - boldone.xx) + λ2 (μ - rr.xx)]
```

Lagrange function for the problem of 7 stocks:

In[164]:=
```
Clear[x, xx, λ1, λ2];
xx = Array[x, {Length[expectedreturns]}];
ones = Table[1, {Length[expectedreturns]}];
lagrange7 = lagrange[xx, expectedreturns, covariancematrixofreturns, λ1, λ2, μ] // Simplify
```

Out[167]=
```
1.22172 x[1]^2 + 1.84727 x[1] x[2] + 2.16234 x[2]^2 + 0.848831 x[1] x[3] +
 1.49482 x[2] x[3] + 2.22362 x[3]^2 + 2.06649 x[1] x[4] + 3.46798 x[2] x[4] +
 1.51603 x[3] x[4] + 6.48492 x[4]^2 + 0.453077 x[1] x[5] + 1.04532 x[2] x[5] +
 0.713154 x[3] x[5] + 0.667194 x[4] x[5] + 0.441274 x[5]^2 + 0.39021 x[1] x[6] +
 0.169168 x[2] x[6] + 1.80391 x[3] x[6] + 0.394081 x[4] x[6] + 0.080174 x[5] x[6] +
 3.885 x[6]^2 + λ1 (1 - 1. x[1] - 1. x[2] - 1. x[3] - 1. x[4] - 1. x[5] - 1. x[6] - 1. x[7]) +
 λ2 (μ - 0.138606 x[1] + 0.184389 x[2] + 0.283755 x[3] + 0.618576 x[4] - 0.0191205 x[5] -
    0.126253 x[6] + 0.0318733 x[7]) + 1.81755 x[1] x[7] + 1.90484 x[2] x[7] +
 1.45582 x[3] x[7] + 3.52243 x[4] x[7] + 0.44031 x[5] x[7] + 2.23644 x[6] x[7] + 2.53456 x[7]^2
```

The gradient and the derivatives with respect to $\lambda1$ and $\lambda2$

In[168]:=
```
(dxx = D[lagrange7, {xx}]) // TableForm
dλ1 = D[lagrange7, λ1]
dλ2 = D[lagrange7, λ2]
```

Out[168]//TableForm=
```
-1. λ1 - 0.138606 λ2 + 2.44344 x[1] + 1.84727 x[2] + 0.848831 x[3] + 2.06649 x[4] + 0.453077 x[5] + 0
-1. λ1 + 0.184389 λ2 + 1.84727 x[1] + 4.32468 x[2] + 1.49482 x[3] + 3.46798 x[4] + 1.04532 x[5] + 0.1
-1. λ1 + 0.283755 λ2 + 0.848831 x[1] + 1.49482 x[2] + 4.44724 x[3] + 1.51603 x[4] + 0.713154 x[5] + 1
-1. λ1 + 0.618576 λ2 + 2.06649 x[1] + 3.46798 x[2] + 1.51603 x[3] + 12.9698 x[4] + 0.667194 x[5] + 0.
-1. λ1 - 0.0191205 λ2 + 0.453077 x[1] + 1.04532 x[2] + 0.713154 x[3] + 0.667194 x[4] + 0.882548 x[5]
-1. λ1 - 0.126253 λ2 + 0.39021 x[1] + 0.169168 x[2] + 1.80391 x[3] + 0.394081 x[4] + 0.080174 x[5] +
-1. λ1 + 0.0318733 λ2 + 1.81755 x[1] + 1.90484 x[2] + 1.45582 x[3] + 3.52243 x[4] + 0.44031 x[5] + 2.
```

Out[169]=
```
1 - 1. x[1] - 1. x[2] - 1. x[3] - 1. x[4] - 1. x[5] - 1. x[6] - 1. x[7]
```

Out[170]=
```
μ - 0.138606 x[1] + 0.184389 x[2] + 0.283755 x[3] +
 0.618576 x[4] - 0.0191205 x[5] - 0.126253 x[6] + 0.0318733 x[7]
```

Construction of the system of $N + 2$ equations (N being the number of assets in question), here $7 + 2 = 9$. Still μ is not specified.

```
In[171]:= equations = {Thread[dxx == 0], dλ1 == 0, dλ2 == 0} // Flatten

Out[171]= {-1. λ1 - 0.138606 λ2 + 2.44344 x[1] + 1.84727 x[2] +
    0.848831 x[3] + 2.06649 x[4] + 0.453077 x[5] + 0.39021 x[6] + 1.81755 x[7] == 0,
  -1. λ1 + 0.184389 λ2 + 1.84727 x[1] + 4.32468 x[2] + 1.49482 x[3] +
    3.46798 x[4] + 1.04532 x[5] + 0.169168 x[6] + 1.90484 x[7] == 0,
  -1. λ1 + 0.283755 λ2 + 0.848831 x[1] + 1.49482 x[2] + 4.44724 x[3] +
    1.51603 x[4] + 0.713154 x[5] + 1.80391 x[6] + 1.45582 x[7] == 0,
  -1. λ1 + 0.618576 λ2 + 2.06649 x[1] + 3.46798 x[2] + 1.51603 x[3] +
    12.9698 x[4] + 0.667194 x[5] + 0.394081 x[6] + 3.52243 x[7] == 0,
  -1. λ1 - 0.0191205 λ2 + 0.453077 x[1] + 1.04532 x[2] + 0.713154 x[3] +
    0.667194 x[4] + 0.882548 x[5] + 0.080174 x[6] + 0.44031 x[7] == 0,
  -1. λ1 - 0.126253 λ2 + 0.39021 x[1] + 0.169168 x[2] + 1.80391 x[3] +
    0.394081 x[4] + 0.080174 x[5] + 7.76999 x[6] + 2.23644 x[7] == 0,
  -1. λ1 + 0.0318733 λ2 + 1.81755 x[1] + 1.90484 x[2] + 1.45582 x[3] +
    3.52243 x[4] + 0.44031 x[5] + 2.23644 x[6] + 5.06911 x[7] == 0,
  1 - 1. x[1] - 1. x[2] - 1. x[3] - 1. x[4] - 1. x[5] - 1. x[6] - 1. x[7] == 0,
  μ - 0.138606 x[1] + 0.184389 x[2] + 0.283755 x[3] +
    0.618576 x[4] - 0.0191205 x[5] - 0.126253 x[6] + 0.0318733 x[7] == 0}

In[172]:= solution = Solve[equations, {xx, λ1, λ2} // Flatten]

Out[172]= {{x[1] → 0.108128 + 1.46849 μ, x[2] → -0.101047 - 0.619178 μ, x[3] → 0.0691962 - 0.96483 μ,
   x[4] → 0.058519 - 0.593025 μ, x[5] → 0.809906 + 0.447827 μ, x[6] → 0.0536972 + 0.261999 μ,
   x[7] → 0.00160004 - 0.00128347 μ, λ1 → 0.768112 - 0.866432 μ, λ2 → -0.866432 + 11.321 μ}}
```

Optimal portfolio for an arbitrary μ:

```
In[173]:= optimalportfolio = (xx /. solution) // First

Out[173]= {0.108128 + 1.46849 μ, -0.101047 - 0.619178 μ, 0.0691962 - 0.96483 μ, 0.058519 - 0.593025 μ,
  0.809906 + 0.447827 μ, 0.0536972 + 0.261999 μ, 0.00160004 - 0.00128347 μ}
```

Check that the total investment is actually equal 1 (i. e. $\sum_{n=1}^{N} x_n = 1$):

```
In[174]:= Total[optimalportfolio]

Out[174]= 1. - 1.55431 × 10^-15 μ
```

Now some concrete examples. The following table shows $\left(\mu, r_p, \sqrt{\operatorname{var}\rho_p}, \boldsymbol{x}\right)$:

In[195]:=
```
Table[{"μ:", m, " Achieved", expectedreturns.optimalportfolio /. μ → m, "\n", "Risk: ",
   √(optimalportfolio.covariancematrixofreturns.optimalportfolio) /. μ → m,
   "Portfolio: \n\n\n", optimalportfolio /. μ → m, "\n\n"},
  {m, 0.04, 0.14, 0.02}] // TableForm
```

Out[195]//TableForm=

μ:	0.04	Achieved	0.04	Risk:	0.846706	Portfolio:	0.166868 −0.125814 0.030603 0.034798 0.827819 0.0641772 0.0015487
μ:	0.06	Achieved	0.06	Risk:	0.839581	Portfolio:	0.196237 −0.138197 0.0113064 0.0229375 0.836776 0.0694171 0.0015230
μ:	0.08	Achieved	0.08	Risk:	0.837817	Portfolio:	0.225607 −0.150581 −0.007990 0.011077 0.845732 0.0746571 0.0014973
μ:	0.1	Achieved	0.1	Risk:	0.841448	Portfolio:	0.254977 −0.162964 −0.027286 −0.000783 0.854689 0.0798971 0.0014716
μ:	0.12	Achieved	0.12	Risk:	0.850406	Portfolio:	0.284347 −0.175348 −0.046583 −0.012644 0.863645 0.0851371 0.0014460
μ:	0.14	Achieved	0.14	Risk:	0.864524	Portfolio:	0.313717 −0.187731 −0.06588 −0.024504 0.872602 0.0903771 0.0014203

Remind the expected returns:

In[177]:=
```
expectedreturns // TableForm
```

Out[177]//TableForm=

```
0.138606
−0.184389
−0.283755
−0.618576
0.0191205
0.126253
−0.0318733
```

We see that short sales appear in all minimum-variance portfolios. With short sales allowed, the optimal portfolio may have the expected return higher than the maximum expected return of any individual asset in question.

Markowitz portfolio, short sales not allowed

This is the case of quadratic programming. We make use of the built - in function NMinimize. Notation as in the previous subsubsection. The error message (in red) relates to the last case where the prescribed return is 0.14 and it exceeds the maximum of available returns. We also see that with the inequality in the wealth condition, the total health is not completely spent.

In[205]:=
```
Table[{"Prescribed return: ", μ, " Portfolio: ",
  xx /. (NMinimize[{xx.covariancematrixofreturns.xx, xx.ones ≤ 1,
      xx.expectedreturns ≥ μ, Thread[xx ≥ 0]} // Flatten, xx] // Last),
  "\n\n"}, {μ, 0.04, 0.14, 0.02}] // TableForm
```

NMinimize::nsol :
There are no points that satisfy the constraints {≪10≫ + 0.0318733 x[7] ≤ 0, −1 + ≪6≫ + x[7] ≤ 0}. ≫

Out[205]//TableForm=

```
Prescribed return:   0.04   Portfolio:   0.235488
                                         0.
                                         0.
                                         0.
                                         0.
                                         0.0582951
                                         0.
Prescribed return:   0.06   Portfolio:   0.353231
                                         0.
                                         0.
                                         0.
                                         0.
                                         0.0874446
                                         0.
Prescribed return:   0.08   Portfolio:   0.470977
                                         0.
                                         0.
                                         0.
                                         0.
                                         0.116589
                                         0.
Prescribed return:   0.1    Portfolio:   0.588721
                                         0.
                                         0.
                                         0.
                                         0.
                                         0.145738
                                         0.
Prescribed return:   0.12   Portfolio:   0.706466
                                         0.
                                         0.
                                         0.
                                         0.
                                         0.174884
                                         0.
Prescribed return:   0.14   Portfolio:   Indeterminate
                                         Indeterminate
                                         Indeterminate
                                         Indeterminate
                                         Indeterminate
                                         Indeterminate
                                         Indeterminate
```

So if you are not too ambitious, you even do not spend your whole wealth. Minimizing risk is the prevailing criterion. If you really want to spend all of your money and you are also willing to avoid the problems with the equality constraints, you may allow your investment to be in a very narrow interval, [0.9999999, 1], say:

```
In[239]:= Print["Portfolio: ",
  x689 = xx /. (NMinimize[{xx.covariancematrixofreturns.xx, 0.9999999 ≤ xx.ones ≤ 1,
         xx.expectedreturns ≥ 0.12, Thread[xx ≥ 0]} // Flatten,
       xx] // Last), "with the sum of weights: ",
  Total[
   x689]]
```

```
Portfolio: {0.684394, 0., 0., 0., 0.137281, 0.178325, 0.}with the sum of weights: 1.
```

Compare with the above inequality constraint: the total wealth of 1 is not spent.

```
In[240]:= Print["Portfolio: ", x689 = xx /.
     (NMinimize[{xx.covariancematrixofreturns.xx, xx.ones ≤ 1, xx.expectedreturns ≥ 0.12,
         Thread[xx ≥ 0]} // Flatten, xx] // Last), "with the sum of weights: ",
  Total[
   x689]]
```

```
Portfolio: {0.706466, 0., 0., 0., 0., 0.174884, 0.}with the sum of weights: 0.88135
```

There might be problems with the equality constraint. In some small examples we likely do not meet them.

```
In[216]:= Print["Portfolio with the prescribed return 0.1 is ",
  xx /. (NMinimize[{xx.covariancematrixofreturns.xx, xx.ones == 1,
        xx.expectedreturns ≥ 0.1, Thread[xx ≥ 0]} // Flatten, xx] // Last)]
```

```
Portfolio with the prescribed return 0.1 is {0.540574, 0., 0., 0., 0.307383, 0.152043, 0.}
```

```
In[226]:= Print["Portfolio with the prescribed return 0.13 is ",
  xx /. (NMinimize[{xx.covariancematrixofreturns.xx, xx.ones == 1,
        xx.expectedreturns ≥ 0.13, Thread[xx ≥ 0]} // Flatten, xx] // Last)]
```

```
Portfolio with the prescribed return 0.13 is {0.756317, 0., 0., 0., 0.0522318, 0.191451, 0.}
```

- ### Sharpe ratio, short sales allowed

We use function NMaximize without any constraints, and then just normalize the solution:

```
In[254]:= Module[{solutionSharpeShort},
  solutionSharpeShort = xx /. (NMaximize[expectedreturns.xx / Sqrt[xx.covariancematrixofreturns.xx], xx] // Last);
  Print["The optimal portfolio is\n", solutionSharpeShort / Total[solutionSharpeShort],
   "\n and it really sums to one: ", solutionSharpeShort / Total[solutionSharpeShort] // Total]]
```

```
The optimal portfolio is
{1.40998, -0.649962, -0.786148, -0.467211, 1.20691, 0.285965, 0.000462209}
 and it really sums to one: 1.
```

- ### Sharpe ratio, short sales not allowed

To exclude very small (from numerical point of view zeros) we use the built-in function Chop.

```
In[256]:= Module[{solutionSharpe}, solutionSharpe = solutionSharpe = xx /. (NMaximize[
     {expectedreturns.xx / Sqrt[xx.covariancematrixofreturns.xx], Thread[xx ≥ 0]} // Flatten, xx] // Last);
   Print["The optimal portfolio is\n", solutionSharpe / Total[solutionSharpe] // Chop,
    "\n and it really sums to one: ", solutionSharpe / Total[solutionSharpe] // Total]]
```

```
The optimal portfolio is
{0.801543, 0, 0, 0, 0, 0.198457, 0}
 and it really sums to one: 1.
```