

# 23030130048\_Fatma Ayu Putri Isma\_EMT STATISTIKA

Nama : Fatma Ayu Putri Isma  
NIM : 23030130048  
Kelas : Pendidikan Matematika C 2023

## EMT untuk Statistika

Dalam buku catatan ini, kami mendemonstrasikan plot statistik utama, pengujian, dan distribusi di Euler.

Mari kita mulai dengan beberapa statistik deskriptif. Ini bukan pengantar statistik. Jadi, Anda mungkin memerlukan beberapa latar belakang untuk memahami detailnya.

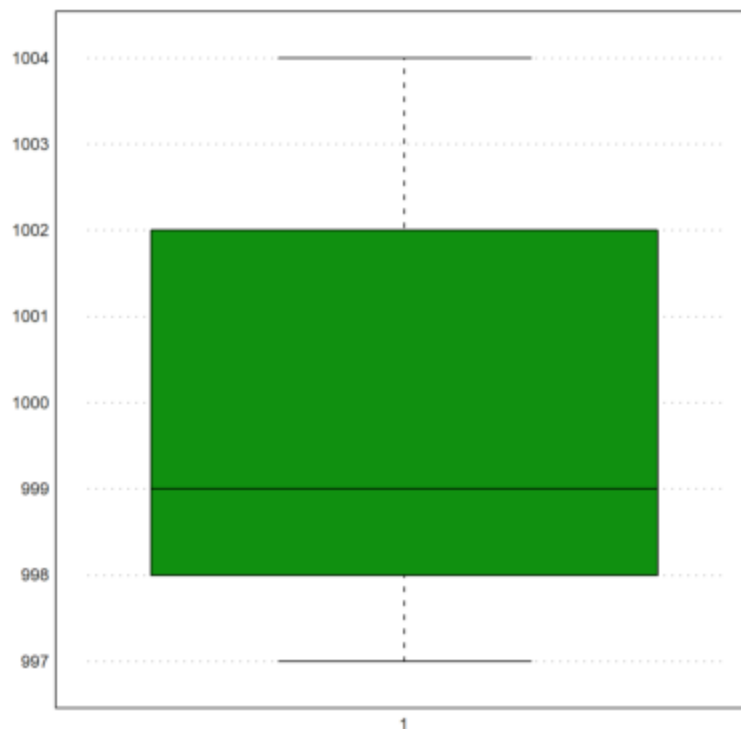
Asumsikan pengukuran berikut. Kami ingin menghitung nilai rata-rata dan standar deviasi yang diukur.

```
>M=[1000,1004,998,997,1002,1001,998,1004,998,997]; ...  
mean(M), dev(M),
```

```
999.9  
2.72641400622
```

Kita dapat memplot plot kotak-dan-kumis untuk data. Dalam kasus kami tidak ada outlier.

```
>boxplot(M) :
```



Kami menghitung probabilitas bahwa suatu nilai lebih besar dari 1005, dengan asumsi nilai terukur dan distribusi normal.

Semua fungsi untuk distribusi di Euler diakhiri dengan `...dis` dan menghitung distribusi probabilitas kumulatif (CPF).

lateks:  $\text{normaldis}(x,m,d)=\int_{-\infty}^x \frac{1}{d\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{t-m}{d}\right)^2} dt$ .

Kami mencetak hasilnya dalam % dengan akurasi 2 digit menggunakan fungsi cetak.

```
>print((1-normaldis(1005,mean(M),dev(M)))*100,2,unit="%")
```

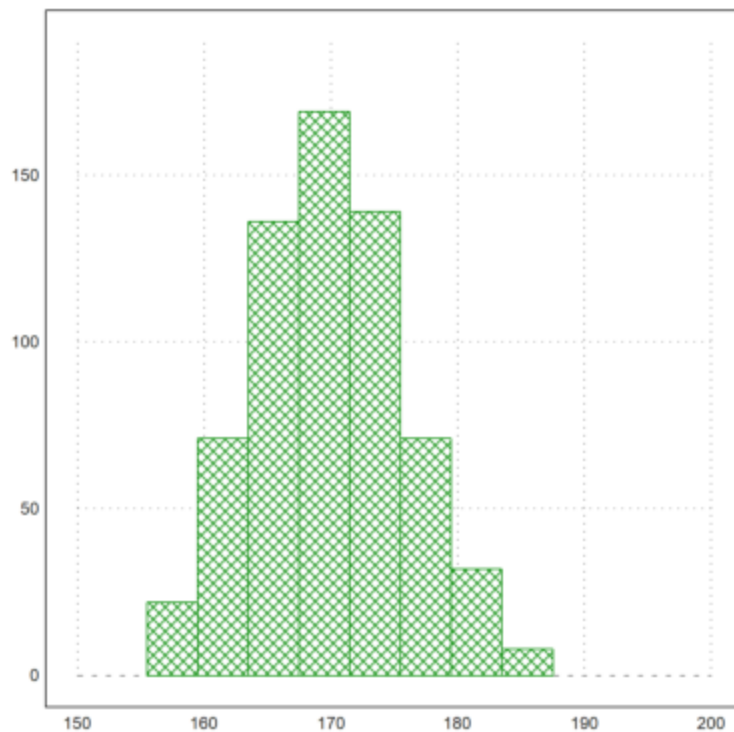
3.07 %

Untuk contoh berikutnya, kami mengasumsikan jumlah pria berikut dalam rentang ukuran yang diberikan.

```
>r=155.5:4:187.5; v=[22,71,136,169,139,71,32,8];
```

Berikut adalah plot distribusinya.

```
>plot2d(r,v,a=150,b=200,c=0,d=190,bar=1,style="\/"):
```



Kita bisa memasukkan data mentah tersebut ke dalam sebuah tabel.

Tabel adalah metode untuk menyimpan data statistik. Tabel kita harus berisi tiga kolom: Awal jangkauan, akhir jangkauan, jumlah orang dalam jangkauan.

Tabel dapat dicetak dengan header. Kami menggunakan vektor string untuk mengatur header.

```
>T:=r[1:8]' | r[2:9]' | v'; writetable(T,labc=["from","to","count"])
```

from	to	count
155.5	159.5	22
159.5	163.5	71
163.5	167.5	136

167.5	171.5	169
171.5	175.5	139
175.5	179.5	71
179.5	183.5	32
183.5	187.5	8

Jika kita membutuhkan nilai rata-rata dan statistik lain dari ukuran, kita perlu menghitung titik tengah rentang. Kita dapat menggunakan dua kolom pertama dari tabel kita untuk ini.

Symbol "|" digunakan untuk memisahkan kolom, fungsi "writetable" digunakan untuk menulis tabel, dengan opsi "labc" adalah untuk menentukan header kolom.

```
>(T[,1]+T[,2])/2 // the midpoint of each interval
```

```
157.5
161.5
165.5
169.5
173.5
177.5
181.5
185.5
```

Tetapi lebih mudah, untuk melipat rentang dengan vektor [1/2.1/2].

```
>M=fold(r,[0.5,0.5])
```

```
[157.5, 161.5, 165.5, 169.5, 173.5, 177.5, 181.5, 185.5]
```

Sekarang kita dapat menghitung mean dan deviasi sampel dengan frekuensi yang diberikan.

```
>{m,d}=meandev(M,v); m, d,
```

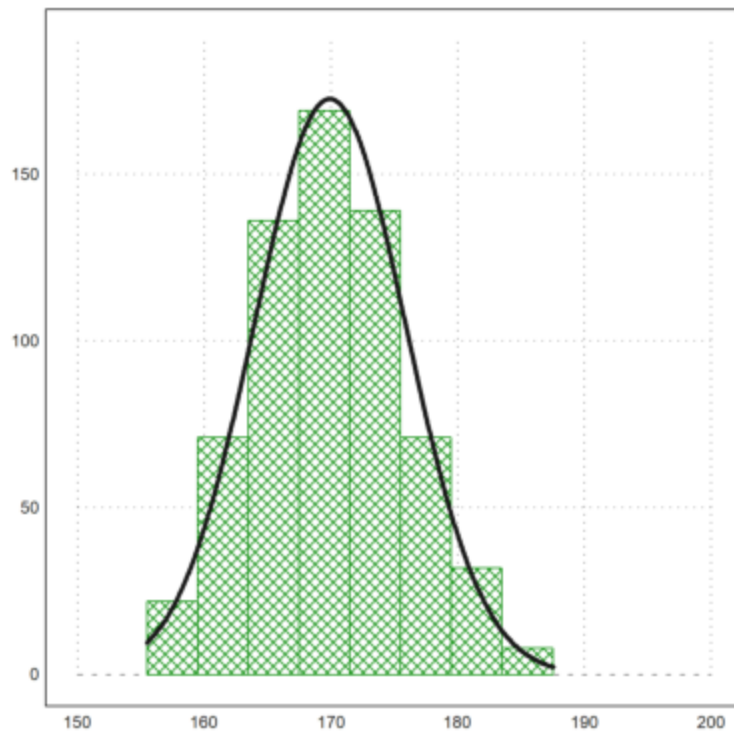
```
169.901234568
5.98912964449
```

Mari kita tambahkan distribusi normal dari nilai-nilai ke plot batang di atas. Rumus untuk distribusi normal dengan mean m dan standar deviasi d adalah:

lateks:  $y = \frac{1}{d\sqrt{2\pi}} e^{-\frac{(x-m)^2}{2d^2}}$ .

Karena nilainya antara 0 dan 1, untuk memplotnya pada bar plot harus dikalikan dengan 4 kali jumlah total data.

```
> plot2d("qnormal(x,m,d)*sum(v)*4", ...
  xmin=min(r),xmax=max(r),thickness=3,add=1):
```



## Meja

Di direktori notebook ini Anda menemukan file dengan tabel. Data tersebut mewakili hasil survei. Berikut adalah empat baris pertama dari file tersebut. Data berasal dari buku online Jerman "Einführung in die Statistik mit R" oleh A. Handl.

```
>printfile("table.dat", 4);
```

```
Person Sex Age Titanic Evaluation Tip Problem
1 m 30 n . 1.80 n
2 f 23 y g 1.80 n
3 f 26 y g 1.80 y
```

Tabel berisi 7 kolom angka atau token (string). Kami ingin membaca tabel dari file. Pertama, kami menggunakan terjemahan kami sendiri untuk token.

Untuk ini, kami mendefinisikan set token. Fungsi `strtokens()` mendapatkan vektor string token dari string yang diberikan.

```
>mf:=["m","f"]; yn:=["y","n"]; ev:=strtokens("g vg m b vb");
```

Sekarang kita membaca tabel dengan terjemahan ini.

Argumen `tok2`, `tok4` dll. adalah terjemahan dari kolom tabel. Argumen ini tidak ada dalam daftar parameter `readtable()`, jadi Anda harus menyediakannya dengan `:=`.

```
>{MT,hd}=readtable("table.dat", tok2:=mf, tok4:=yn, tok5:=ev, tok7:=yn);
>load over statistics;
```

Untuk mencetak, kita perlu menentukan set token yang sama. Kami mencetak empat baris pertama saja.

```
>writetable(MT[1:4], labc=hd, wc=5, tok2:=mf, tok4:=yn, tok5:=ev, tok7:=yn);
```

Person	Sex	Age	Titanic	Evaluation	Tip	Problem
1	m	30	n	.	1.8	n
2	f	23	y	g	1.8	n
3	f	26	y	g	1.8	y
4	m	33	n	.	2.8	n

Titik "." mewakili nilai-nilai, yang tidak tersedia.

Jika kita tidak ingin menentukan token untuk terjemahan terlebih dahulu, kita hanya perlu menentukan, kolom mana yang berisi token dan bukan angka.

```
>ctok=[2,4,5,7]; {MT,hd,tok}=readtable("table.dat",ctok=ctok);
```

Fungsi readtable() sekarang mengembalikan satu set token.

```
>tok
```

```
m
n
f
y
g
vg
```

Tabel berisi entri dari file dengan token yang diterjemahkan ke angka.

String khusus NA="." ditafsirkan sebagai "Tidak Tersedia", dan mendapatkan NAN (bukan angka) dalam tabel. Terjemahan ini dapat diubah dengan parameter NA, dan NAval.

```
>MT[1]
```

```
[1, 1, 30, 2, NAN, 1.8, 2]
```

Berikut isi tabel dengan angka yang belum diterjemahkan.

```
>writetable(MT,wc=5)
```

1	1	30	2	.	1.8	2
2	3	23	4	5	1.8	2
3	3	26	4	5	1.8	4
4	1	33	2	.	2.8	2
5	1	37	2	.	1.8	2
6	1	28	4	5	2.8	4
7	3	31	4	6	2.8	2
8	1	23	2	.	0.8	2
9	3	24	4	6	1.8	4
10	1	26	2	.	1.8	2
11	3	23	4	6	1.8	4
12	1	32	4	5	1.8	2
13	1	29	4	6	1.8	4
14	3	25	4	5	1.8	4
15	3	31	4	5	0.8	2
16	1	26	4	5	2.8	2
17	1	37	2	.	3.8	2
18	1	38	4	5	.	2
19	3	29	2	.	3.8	2

20	3	28	4	6	1.8	2
21	3	28	4	1	2.8	4
22	3	28	4	6	1.8	4
23	3	38	4	5	2.8	2
24	3	27	4	1	1.8	4
25	1	27	2	.	2.8	4

Untuk kenyamanan, Anda dapat memasukkan output `readtable()` ke dalam daftar.

```
>Table={{readtable("table.dat",ctok=ctok)}};
```

Menggunakan kolom token yang sama dan token yang dibaca dari file, kita dapat mencetak tabel. Kita dapat menentukan `ctok`, `tok`, dll. Atau menggunakan daftar Tabel.

```
>writetable(Table,ctok=ctok,wc=5);
```

Person	Sex	Age	Titanic	Evaluation	Tip	Problem
1	m	30	n	.	1.8	n
2	f	23	y	g	1.8	n
3	f	26	y	g	1.8	y
4	m	33	n	.	2.8	n
5	m	37	n	.	1.8	n
6	m	28	y	g	2.8	y
7	f	31	y	vg	2.8	n
8	m	23	n	.	0.8	n
9	f	24	y	vg	1.8	y
10	m	26	n	.	1.8	n
11	f	23	y	vg	1.8	y
12	m	32	y	g	1.8	n
13	m	29	y	vg	1.8	y
14	f	25	y	g	1.8	y
15	f	31	y	g	0.8	n
16	m	26	y	g	2.8	n
17	m	37	n	.	3.8	n
18	m	38	y	g	.	n
19	f	29	n	.	3.8	n
20	f	28	y	vg	1.8	n
21	f	28	y	m	2.8	y
22	f	28	y	vg	1.8	y
23	f	38	y	g	2.8	n
24	f	27	y	m	1.8	y
25	m	27	n	.	2.8	y

Fungsi `tablecol()` mengembalikan nilai kolom tabel, melewati setiap baris dengan nilai `NAN("."` dalam file), dan indeks kolom, yang berisi nilai-nilai ini.

```
>{c,i}=tablecol(MT,[5,6]);
```

Kita dapat menggunakan ini untuk mengekstrak kolom dari tabel untuk tabel baru.

```
>j=[1,5,6]; writetable(MT[i,j],labc=hd[j],ctok=[2],tok=tok)
```

Person	Evaluation	Tip
2	g	1.8
3	g	1.8
6	g	2.8
7	vg	2.8
9	vg	1.8

11	vg	1.8
12	g	1.8
13	vg	1.8
14	g	1.8
15	g	0.8
16	g	2.8
20	vg	1.8
21	m	2.8
22	vg	1.8
23	g	2.8
24	m	1.8

Tentu saja, kita perlu mengekstrak tabel itu sendiri dari daftar Tabel dalam kasus ini.

```
>MT=Table[1];
```

Tentu saja, kita juga dapat menggunakannya untuk menentukan nilai rata-rata kolom atau nilai statistik lainnya.

```
>mean(tablecol(MT,6))
```

2.175

Fungsi `getstatistics()` mengembalikan elemen dalam vektor, dan jumlahnya. Kami menerapkannya pada nilai "m" dan "f" di kolom kedua tabel kami.

```
>{xu,count}=getstatistics(tablecol(MT,2)); xu, count,
```

```
[1, 3]
[12, 13]
```

Kami dapat mencetak hasilnya dalam tabel baru.

```
>writetable(count',labr=tok[xu])
```

```
  m      12
  f      13
```

Fungsi `selecttable()` mengembalikan tabel baru dengan nilai dalam satu kolom yang dipilih dari vektor indeks. Pertama kita mencari indeks dari dua nilai kita di tabel token.

```
>v:=indexof(tok,["g","vg"])
```

```
[5, 6]
```

Sekarang kita dapat memilih baris tabel, yang memiliki salah satu nilai dalam v di baris ke-5.

```
>MT1:=MT[selectrows(MT,5,v)]; i:=sortedrows(MT1,5);
```

Sekarang kita dapat mencetak tabel, dengan nilai yang diekstrak dan diurutkan di kolom ke-5.

```
>writetable(MT1[i],labc=hd,ctok=ctok,tok=tok,wc=7);
```





Tabel dapat ditulis ke file.

```
>filename="test.dat"; ...
writetable(count,labr=tok[xu1],labc=tok[xu2],file=filename);
```

Kemudian kita bisa membaca tabel dari file tersebut.

```
>{MT2,hd,tok2,hdr}=readtable(filename,>clabs,>rlabs); ...
writetable(MT2,labr=hdr,labc=hd)
```

	n	y
m	7	5
f	1	12

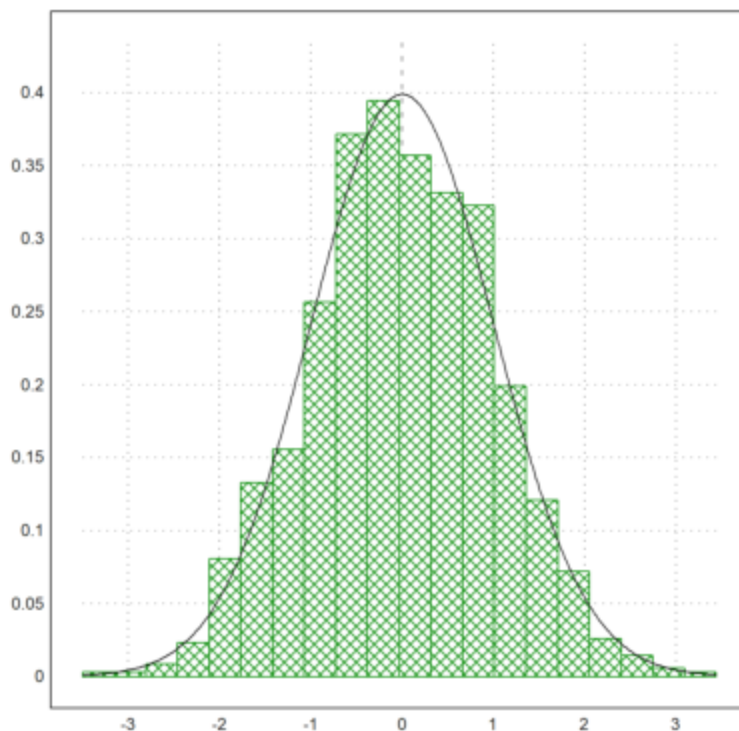
Dan hapus file tersebut.

```
>fileremove(filename);
```

## Distribusi

Dengan plot2d, terdapat metode yang sangat mudah untuk memplot sebaran data eksperimen.

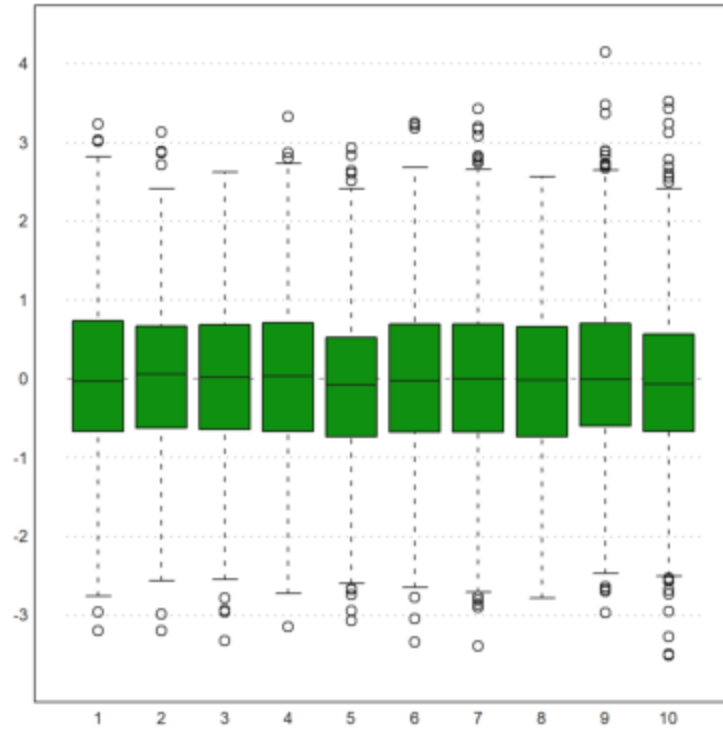
```
>p=normal(1,1000); //1000 random normal-distributed sample p
>plot2d(p,distribution=20,style="\"); // plot the random sample p
>plot2d("qnormal(x,0,1)",add=1): // add the standard normal distribution plot
```



Harap perhatikan perbedaan antara plot batang (sampel) dan kurva normal (distribusi nyata). Masukkan kembali tiga perintah untuk melihat hasil pengambilan sampel lainnya.

Berikut adalah perbandingan 10 simulasi dari 1000 nilai terdistribusi normal menggunakan apa yang disebut plot kotak. Plot ini menunjukkan median, kuartil 25% dan 75%, nilai minimal dan maksimal, dan outlier.

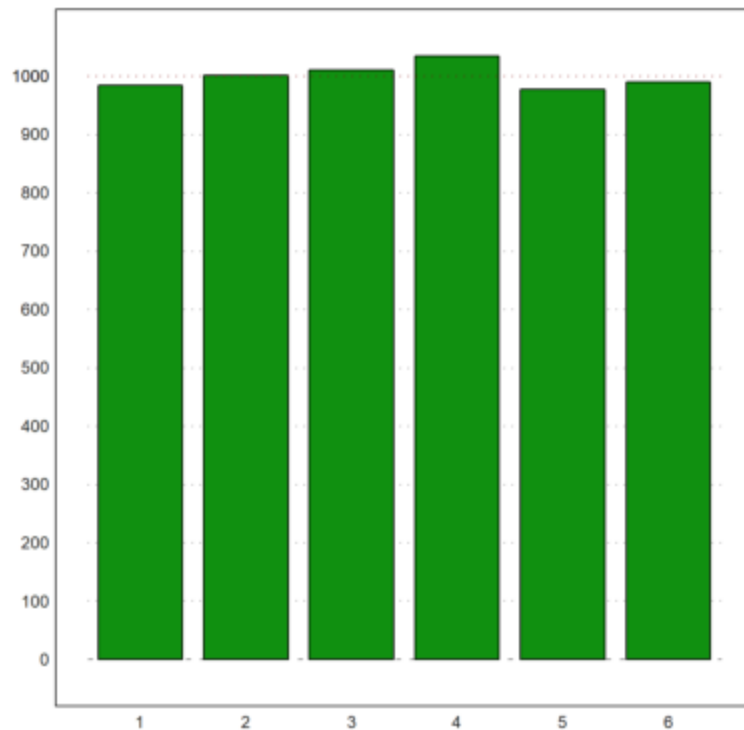
```
>p=normal(10,1000); boxplot(p):
```



Untuk membangkitkan bilangan bulat acak, Euler memiliki intranomial. Mari kita simulasikan lemparan dadu dan plot distribusinya.

Kita menggunakan fungsi `getmultiplicities(v,x)`, yang menghitung seberapa sering elemen `v` muncul di `x`. Kemudian kita memplot hasilnya menggunakan `columnplot()`.

```
>k=intrand(1,6000,6); ...
columnplot(getmultiplicities(1:6,k)); ...
ygrid(1000,color=red):
```



Sementara `inrandom(n,m,k)` mengembalikan bilangan bulat yang terdistribusi secara seragam dari 1 ke k, dimungkinkan untuk menggunakan distribusi bilangan bulat lain yang diberikan dengan `randpint()`.

Dalam contoh berikut, probabilitas untuk 1,2,3 masing-masing adalah 0,4,0,1,0,5.

```
>randpint(1,1000,[0.4,0.1,0.5]); getmultiplicities(1:3,%)
```

```
[378, 102, 520]
```

Euler dapat menghasilkan nilai acak dari lebih banyak distribusi. Coba lihat referensinya.

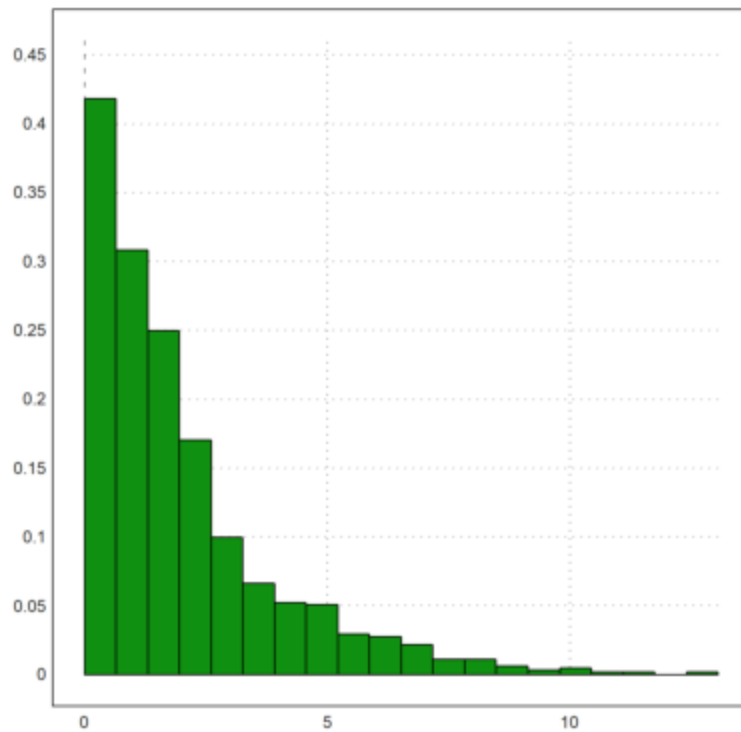
Misalnya, kami mencoba distribusi eksponensial. Variabel acak kontinu  $X$  dikatakan memiliki distribusi eksponensial, jika PDF-nya diberikan oleh

lateks:  $f_X(x)=\lambda e^{-\lambda x}, \text{quad } x>0, \text{quad } \lambda>0,$

dengan parameter

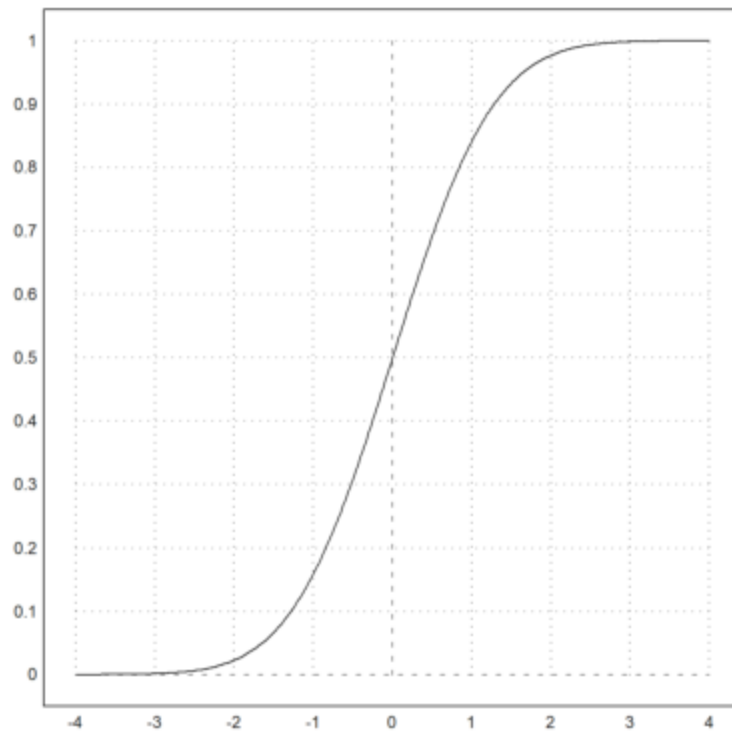
lateks:  $\lambda=\frac{1}{\mu}, \text{quad } \mu \text{ \textit{adalah rata-rata, dan dilambangkan dengan } } X \sim \text{Ekspensial}(\lambda).$

```
>plot2d(randexponential(1,1000,2),>distribution):
```



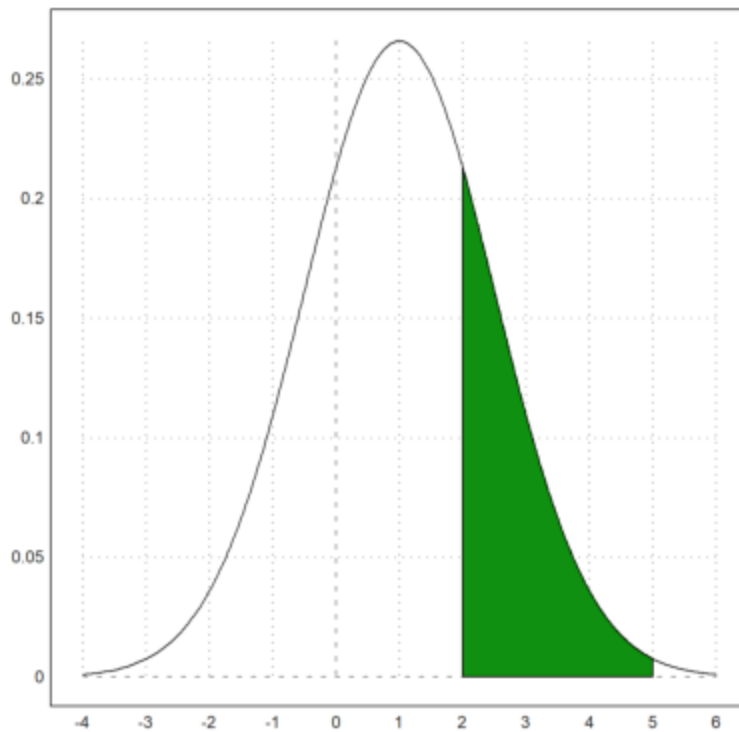
Untuk banyak distribusi, Euler dapat menghitung fungsi distribusi dan inversnya.

```
>plot2d("normaldis",-4,4):
```



Berikut ini adalah salah satu cara untuk memplot kuantil.

```
>plot2d("qnormal(x,1,1.5)",-4,6); ...
plot2d("qnormal(x,1,1.5)",a=2,b=5,>add,>filled):
```



lateks:  $\text{normaldis}(x,m,d)=\int_{-\infty}^x \frac{1}{d\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{t-m}{d})^2} dt$ .  
 Probabilitas untuk berada di area hijau adalah sebagai berikut.

```
>normaldis(5,1,1.5)-normaldis(2,1,1.5)
```

```
0.248662156979
```

Ini dapat dihitung secara numerik dengan integral berikut.

lateks:  $\int_2^5 \frac{1}{1.5\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-1}{1.5})^2} dx$ .

```
>gauss("qnormal(x,1,1.5)",2,5)
```

```
0.248662156979
```

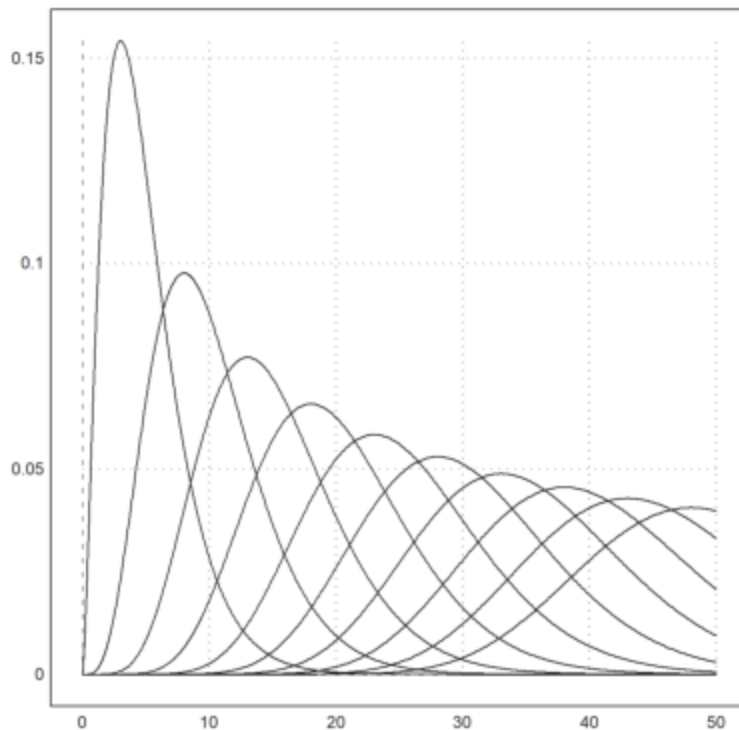
Mari kita bandingkan distribusi binomial dengan distribusi normal mean dan deviasi yang sama. Fungsi `invbindis()` memecahkan interpolasi linier antara nilai bilangan bulat.

```
>invbindis(0.95,1000,0.5), invnormaldis(0.95,500,0.5*sqrt(1000))
```

```
525.516721219  
526.007419394
```

Fungsi `qdis()` adalah kepadatan distribusi chi-kuadrat. Seperti biasa, Euler memetakan vektor ke fungsi ini. Jadi kita mendapatkan plot dari semua distribusi chi-kuadrat dengan derajat 5 sampai 30 dengan mudah dengan cara berikut.

```
>plot2d("qchidis(x,(5:5:50)')",0,50):
```



Euler memiliki fungsi yang akurat untuk mengevaluasi distribusi. Mari kita periksa `chidis()` dengan integral.

Penamaan mencoba untuk konsisten. Misalnya.,

- distribusi chi-kuadrat adalah `chidis()`,
- fungsi kebalikannya adalah `invchidis()`,
- densitasnya adalah `qchidis()`.

Pelengkap distribusi (ekor atas) adalah `chicdis()`.

```
>chidis(1.5,2), integrate("qchidis(x,2)",0,1.5)
```

```
0.527633447259
0.527633447259
```

## Distribusi Diskrit

Untuk menentukan distribusi diskrit Anda sendiri, Anda dapat menggunakan metode berikut.

Pertama kita mengatur fungsi distribusi.

```
>wd = 0 | ((1:6)+[-0.01,0.01,0,0,0,0])/6
```

```
[0, 0.165, 0.335, 0.5, 0.666667, 0.833333, 1]
```

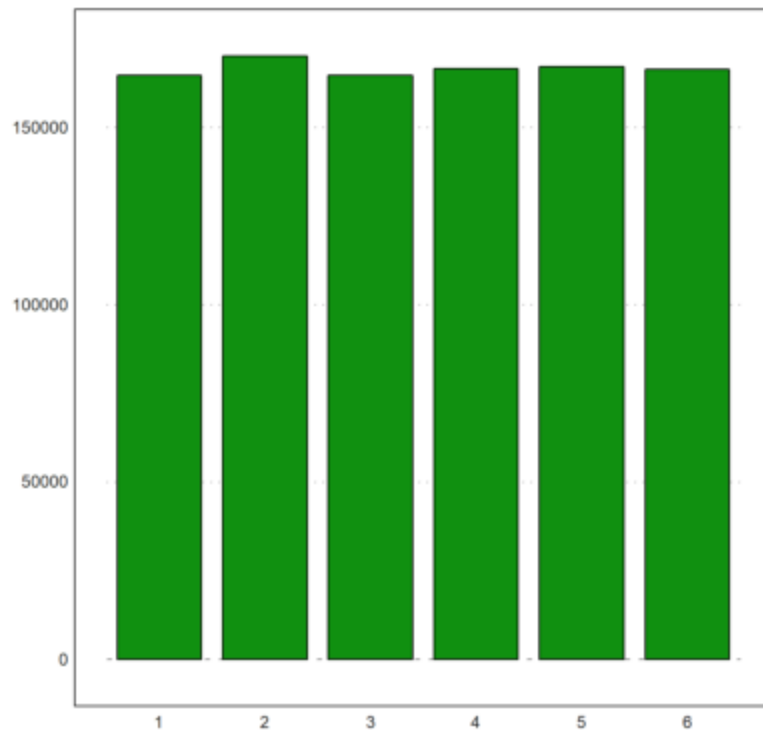
Artinya dengan probabilitas `wd[i+1]-wd[i]` kita menghasilkan nilai acak `i`.

Ini hampir merupakan distribusi yang seragam. Mari kita tentukan generator angka acak untuk ini. Fungsi `find(v,x)` menemukan nilai `x` dalam vektor `v`. Fungsi ini juga berlaku untuk vektor `x`.

```
>function wrongdice (n,m) := find(wd,random(n,m))
```

Kesalahannya sangat halus sehingga kami melihatnya hanya dengan iterasi yang sangat banyak.

```
>columnplot (getmultiplicities (1:6, wrongdice (1, 1000000))):
```



Berikut adalah fungsi sederhana untuk memeriksa distribusi seragam dari nilai 1...K dalam v. Kami menerima hasilnya, jika untuk semua frekuensi

lateks:  $\left|f_i - \frac{1}{K}\right| < \frac{\delta}{\sqrt{n}}$ .

```
>function checkrandom (v, delta=1) ...
  K=max(v); n=cols(v);
  fr=getfrequencies(v, 1:K);
  return max(fr/n-1/K)<delta/sqrt(n);
endfunction
```

Memang fungsi menolak distribusi seragam.

```
>checkrandom(wrongdice(1, 1000000))
```

0

Dan itu menerima generator acak bawaan.

```
>checkrandom(inrandom(1, 1000000, 6))
```

1

Kita dapat menghitung distribusi binomial. Pertama ada binomialsum(), yang mengembalikan probabilitas i atau kurang hit dari n percobaan.

```
>bindis(410,1000,0.4)
```

```
0.751401349654
```

Fungsi Beta terbalik digunakan untuk menghitung interval kepercayaan Clopper-Pearson untuk parameter  $p$ . Level default adalah alfa.

Arti interval ini adalah jika  $p$  berada di luar interval, hasil pengamatan 410 dalam 1000 jarang terjadi.

```
>clopperpearson(410,1000)
```

```
[0.37932, 0.441212]
```

Perintah berikut adalah cara langsung untuk mendapatkan hasil di atas. Tapi untuk  $n$  besar, penjumlahan langsungnya tidak akurat dan lambat.

```
>p=0.4; i=0:410; n=1000; sum(bin(n,i)*p^i*(1-p)^(n-i))
```

```
0.751401349655
```

Omong-omong, `invbinsum()` menghitung kebalikan dari `binomialsum()`.

```
>invbindis(0.75,1000,0.4)
```

```
409.932733047
```

Di Bridge, kami mengasumsikan 5 kartu beredar (dari 52) dengan dua tangan (26 kartu). Mari kita hitung probabilitas distribusi yang lebih buruk dari 3:2 (mis. 0:5, 1:4, 4:1, atau 5:0).

```
>2*hypergeomsum(1,5,13,26)
```

```
0.321739130435
```

Ada juga simulasi distribusi multinomial.

```
>randmultinomial(10,1000,[0.4,0.1,0.5])
```

381	100	519
376	91	533
417	80	503
440	94	466
406	112	482
408	94	498
395	107	498
399	96	505
428	87	485
400	99	501



Untuk memplot data, kami mencoba hasil pemilu Jerman sejak 1990, yang diukur dalam jumlah kursi.

```
>BW := [ ...
1990, 662, 319, 239, 79, 8, 17; ...
1994, 672, 294, 252, 47, 49, 30; ...
1998, 669, 245, 298, 43, 47, 36; ...
2002, 603, 248, 251, 47, 55, 2; ...
2005, 614, 226, 222, 61, 51, 54; ...
2009, 622, 239, 146, 93, 68, 76; ...
2013, 631, 311, 193, 0, 63, 64];
```

Untuk partai, kami menggunakan rangkaian nama.

```
>P:=["CDU/CSU", "SPD", "FDP", "Gr", "Li"];
```

Mari kita cetak persentasenya dengan baik.

Pertama, kami mengekstrak kolom yang diperlukan. Kolom 3 sampai 7 adalah kursi masing-masing partai, dan kolom 2 adalah jumlah kursi. kolom adalah tahun pemilihan.

```
>BT:=BW[, 3:7]; BT:=BT/sum(BT); YT:=BW[, 1]';
```

Kemudian kami mencetak statistik dalam bentuk tabel. Kami menggunakan nama sebagai tajuk kolom, dan tahun sebagai tajuk untuk baris. Lebar default untuk kolom adalah wc=10, tetapi kami lebih memilih hasil yang lebih padat. Kolom akan diperluas untuk label kolom, jika perlu.

```
>writetable(BT*100, wc=6, dc=0, >fixed, labc=P, labr=YT)
```

	CDU/CSU	SPD	FDP	Gr	Li
1990	48	36	12	1	3
1994	44	38	7	7	4
1998	37	45	6	7	5
2002	41	42	8	9	0
2005	37	36	10	8	9
2009	38	23	15	11	12
2013	49	31	0	10	10

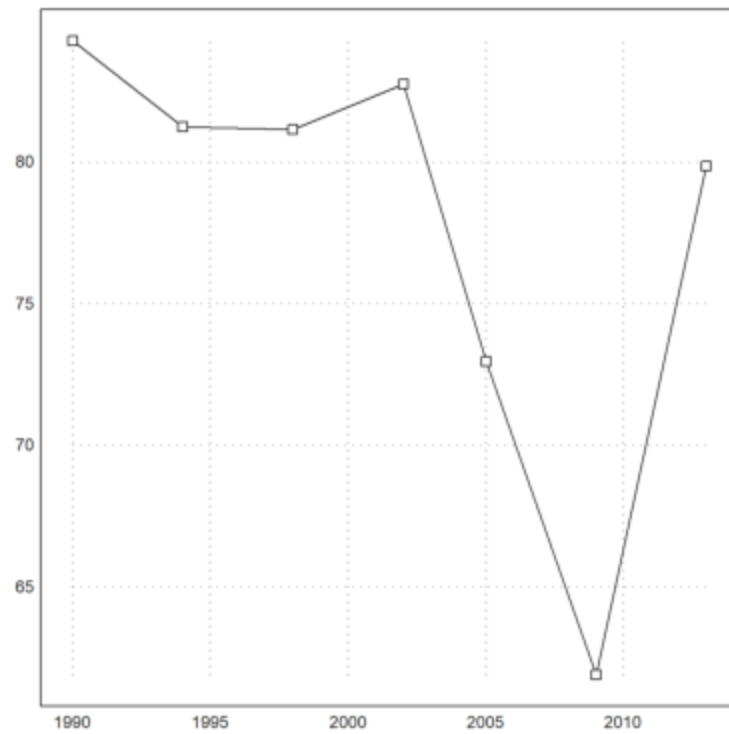
Perkalian matriks berikut mengekstrak jumlah persentase dari dua partai besar yang menunjukkan bahwa partai-partai kecil telah mendapatkan rekaman di parlemen hingga tahun 2009.

```
>BT1:=(BT.[1;1;0;0;0])'*100
```

```
[84.29, 81.25, 81.1659, 82.7529, 72.9642, 61.8971, 79.8732]
```

Ada juga plot statistik sederhana. Kami menggunakannya untuk menampilkan garis dan titik secara bersamaan. Alternatifnya adalah memanggil plot2d dua kali dengan >add.

```
>statplot(YT, BT1, "b"):
```

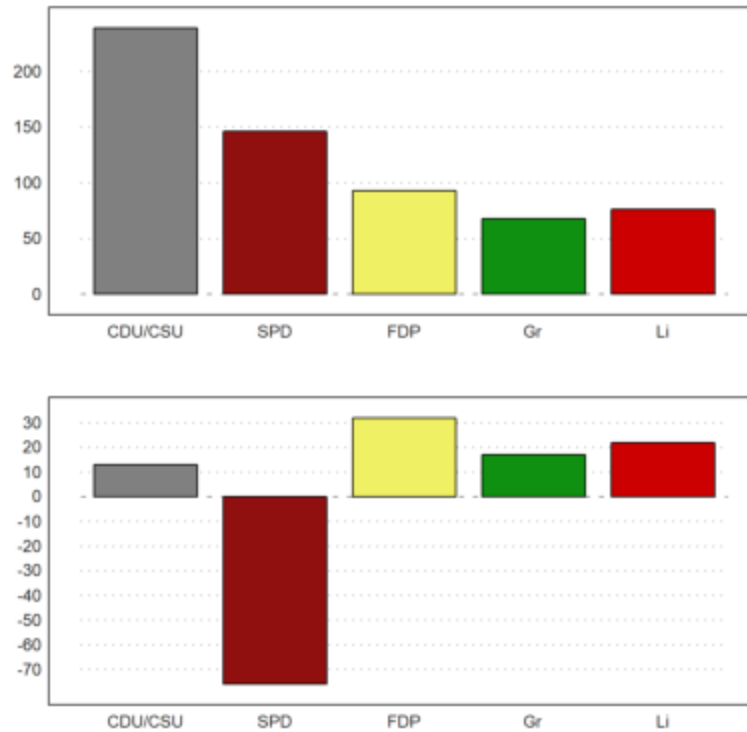


Tentukan beberapa warna untuk masing-masing pihak.

```
>CP:=[rgb(0.5,0.5,0.5),red,yellow,green,rgb(0.8,0,0)];
```

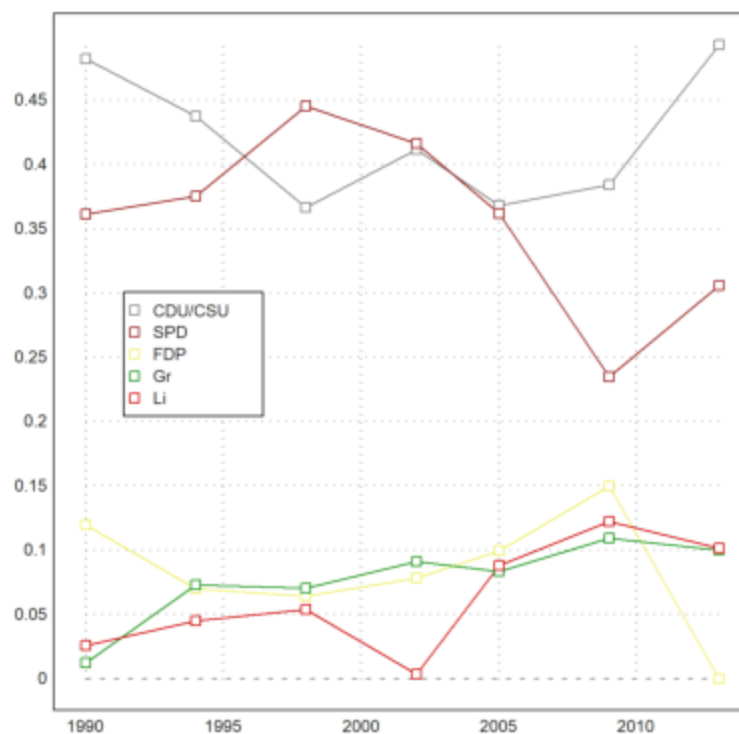
Sekarang kita bisa memplot hasil pemilu 2009 dan mengubahnya menjadi satu plot menggunakan gambar. Kita dapat menambahkan vektor kolom ke setiap plot.

```
>figure(2,1); ...
figure(1); columnsplot(BW[6,3:7],P,color=CP); ...
figure(2); columnsplot(BW[6,3:7]-BW[5,3:7],P,color=CP); ...
figure(0):
```



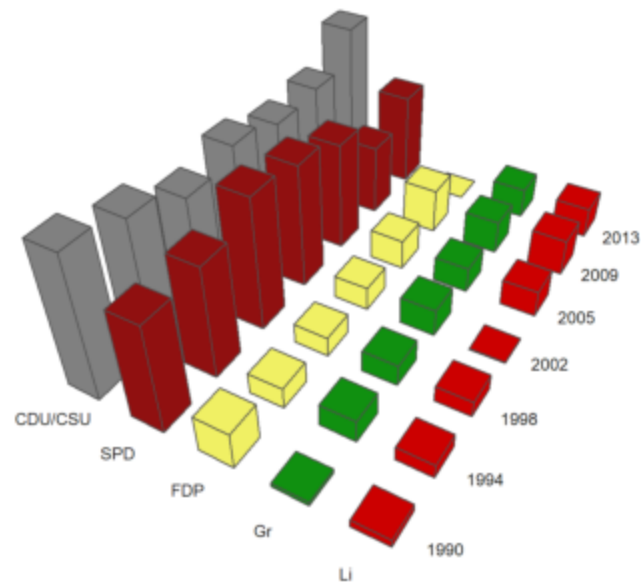
Plot data menggabungkan deretan data statistik dalam satu plot.

```
>J:=BW[,1]'; DP:=BW[,3:7]'; ...
dataplot(YT,BT',color=CP); ...
labelbox(P,colors=CP,styles="[]",>points,w=0.2,x=0.3,y=0.4):
```



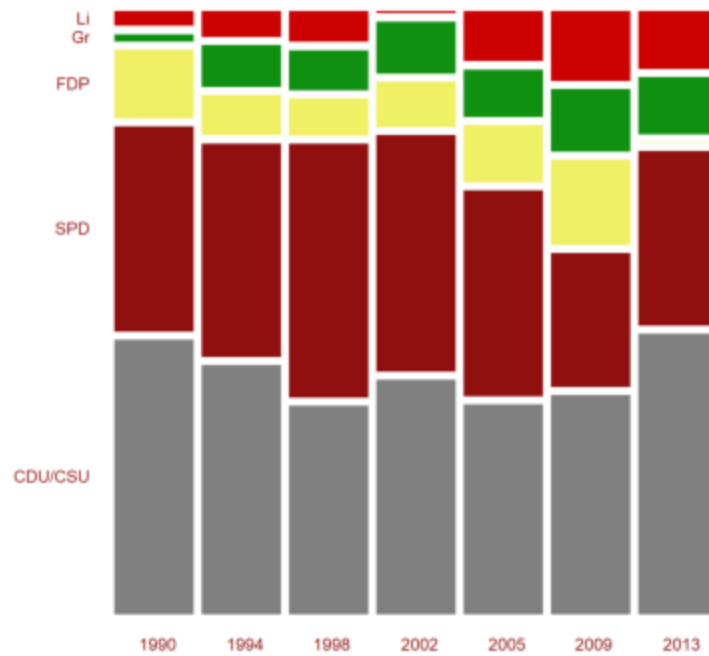
Plot kolom 3D menunjukkan deretan data statistik dalam bentuk kolom. Kami menyediakan label untuk baris dan kolom. sudut adalah sudut pandang.

```
>columnplot3d(BT,scols=P,srows=YT, ...
  angle=30°,ccols=CP) :
```



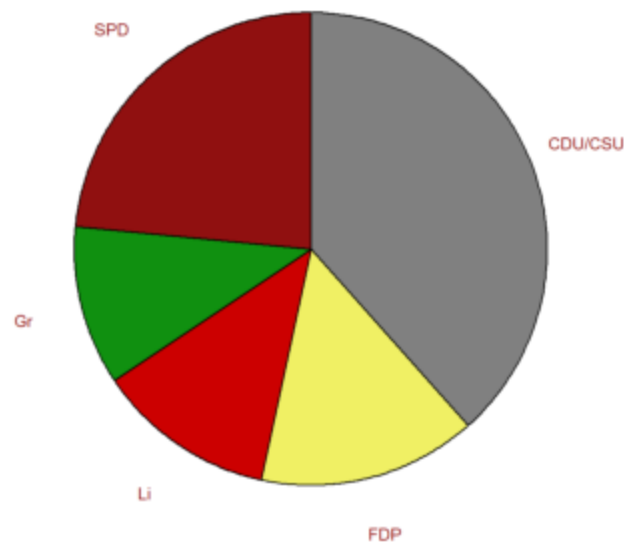
Representasi lainnya adalah plot mozaik. Perhatikan bahwa kolom plot mewakili kolom matriks di sini. Karena panjang label CDU/CSU, kami mengambil jendela yang lebih kecil dari biasanya.

```
>shrinkwindow(>smaller); ...
mosaicplot(BT',srows=YT,scols=P,color=CP,style="#"); ...
shrinkwindow() :
```



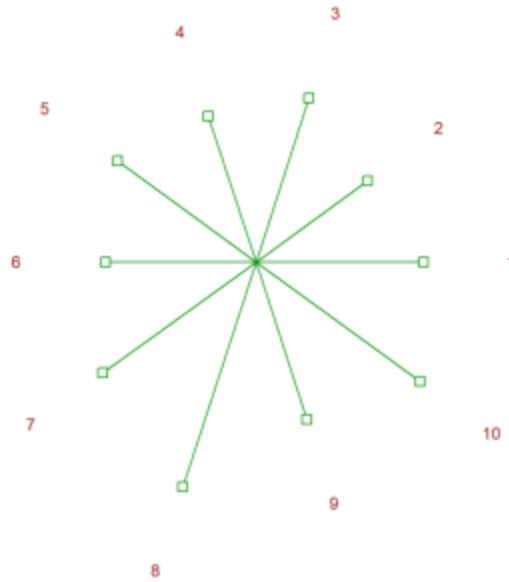
Kita juga bisa membuat diagram lingkaran. Karena hitam dan kuning membentuk koalisi, kami menyusun ulang elemennya.

```
>i=[1,3,5,4,2]; piechart(BW[6,3:7][i],color=CP[i],lab=P[i]):
```



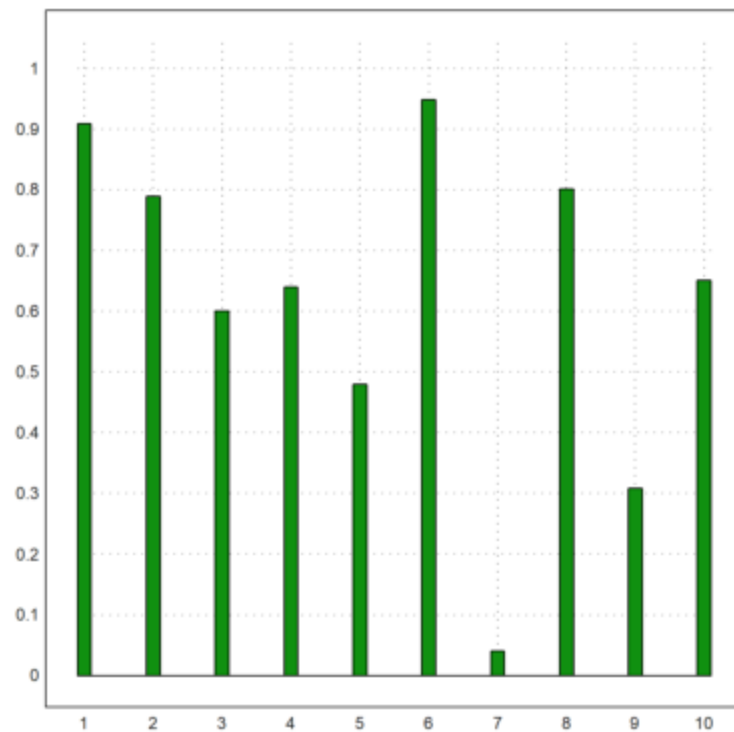
Ini jenis plot lainnya.

```
>starplot(normal(1,10)+4,lab=1:10,>rays):
```



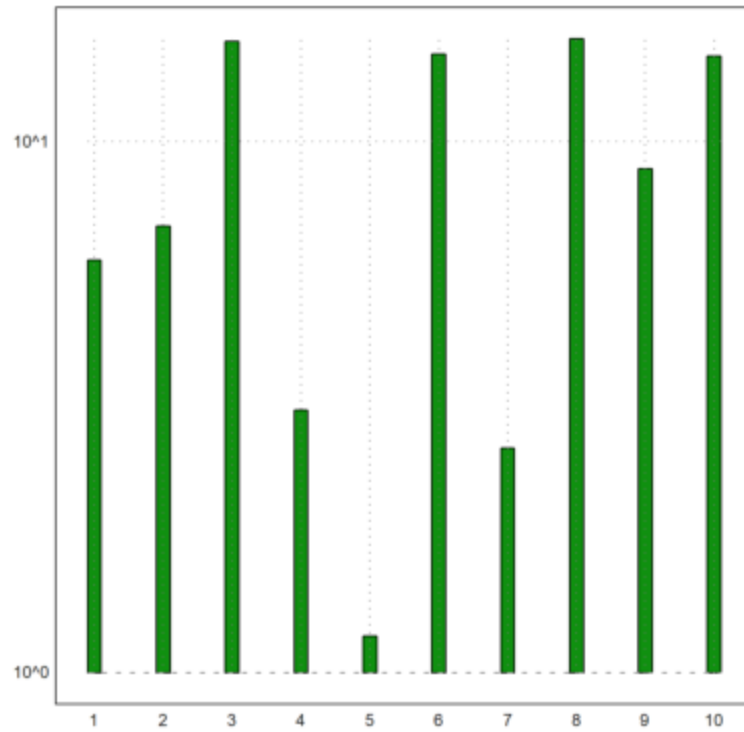
Beberapa plot di plot2d bagus untuk statika. Berikut adalah plot impuls dari data acak, terdistribusi secara seragam di  $[0,1]$ .

```
>plot2d(makeimpulse(1:10, random(1,10)), >bar) :
```



Tetapi untuk data yang terdistribusi secara eksponensial, kita mungkin memerlukan plot logaritmik.

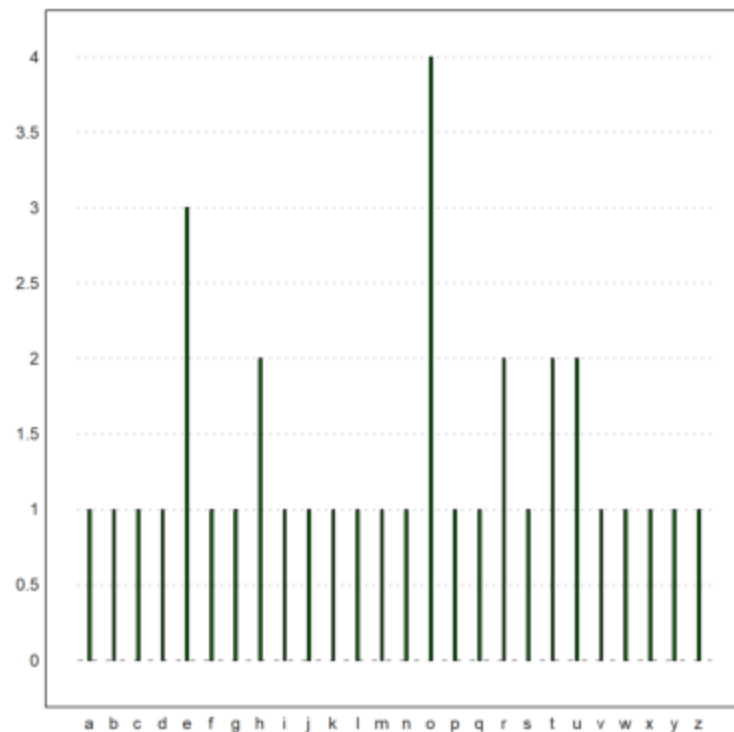
```
>logimpulseplot(1:10, -log(random(1,10))*10) :
```



Fungsi `columnplot()` lebih mudah digunakan, karena hanya membutuhkan vektor nilai. Selain itu, ia dapat mengatur labelnya ke apa pun yang kita inginkan, kita telah mendemonstrasikannya di tutorial ini.

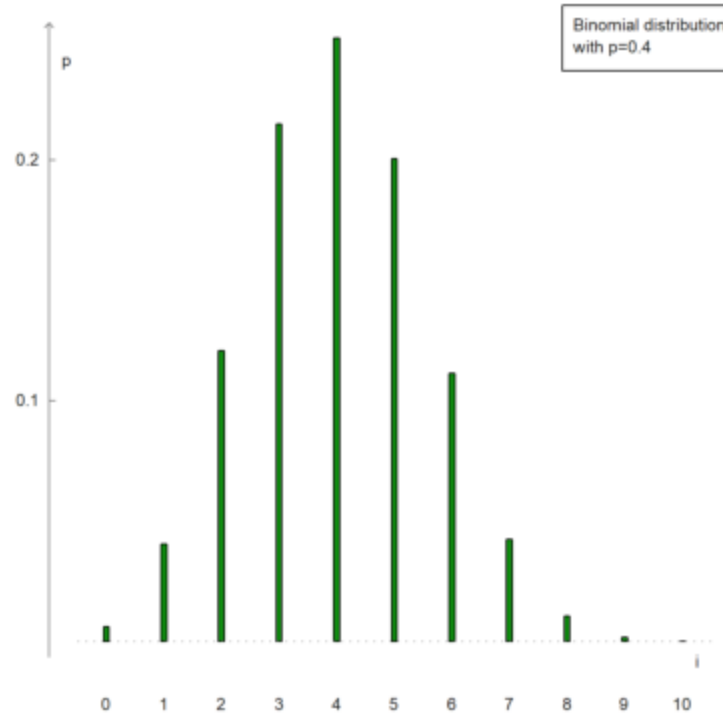
Ini adalah aplikasi lain, di mana kami menghitung karakter dalam sebuah kalimat dan memplot statistik.

```
>v=strtochar("the quick brown fox jumps over the lazy dog"); ...
w=ascii("a"):ascii("z"); x=getmultiplicities(w,v); ...
cw=[]; for k=w; cw=cw|char(k); end; ...
columnplot(x,lab=cw,width=0.05):
```



Dimungkinkan juga untuk mengatur sumbu secara manual.

```
>n=10; p=0.4; i=0:n; x=bin(n,i)*p^i*(1-p)^(n-i); ...
columnsplot(x,lab=i,width=0.05,<frame,<grid); ...
yaxis(0,0:0.1:1,style="->",>left); xaxis(0,style="."); ...
label("p",0,0.25), label("i",11,0); ...
textbox(["Binomial distribution","with p=0.4"]):
```



Berikut ini adalah cara memplot frekuensi bilangan dalam vektor.

Kami membuat vektor bilangan acak bilangan bulat 1 hingga 6.

```
>v:=inrandom(1,10,10)
```

```
[8, 5, 8, 8, 6, 8, 8, 3, 5, 5]
```

Kemudian ekstrak angka unik di v.

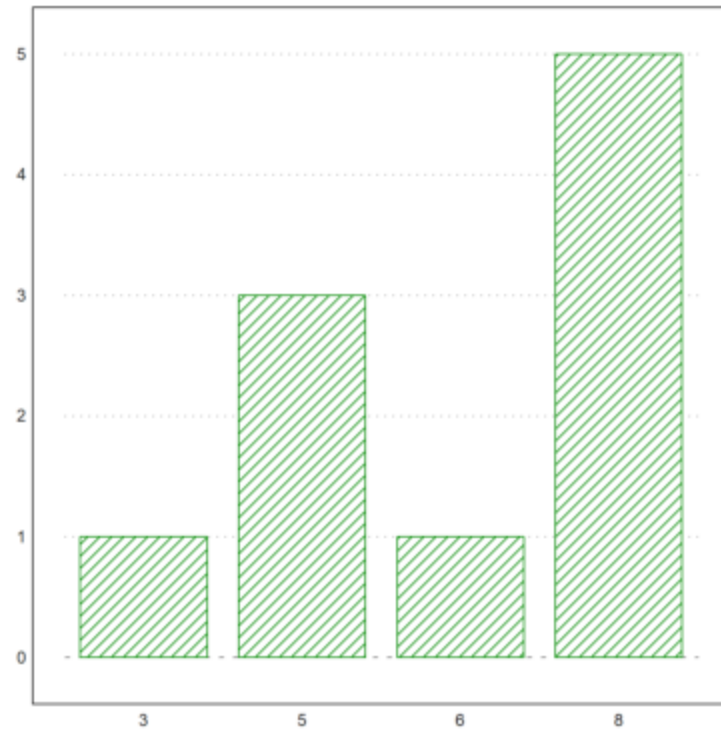
```
>vu:=unique(v)
```

```
[3, 5, 6, 8]
```

Dan plot frekuensi dalam plot kolom.

```
>columnsplot(getmultiplicities(vu,v),lab=vu,style="/"):
```





Kami ingin menunjukkan fungsi untuk distribusi nilai empiris.

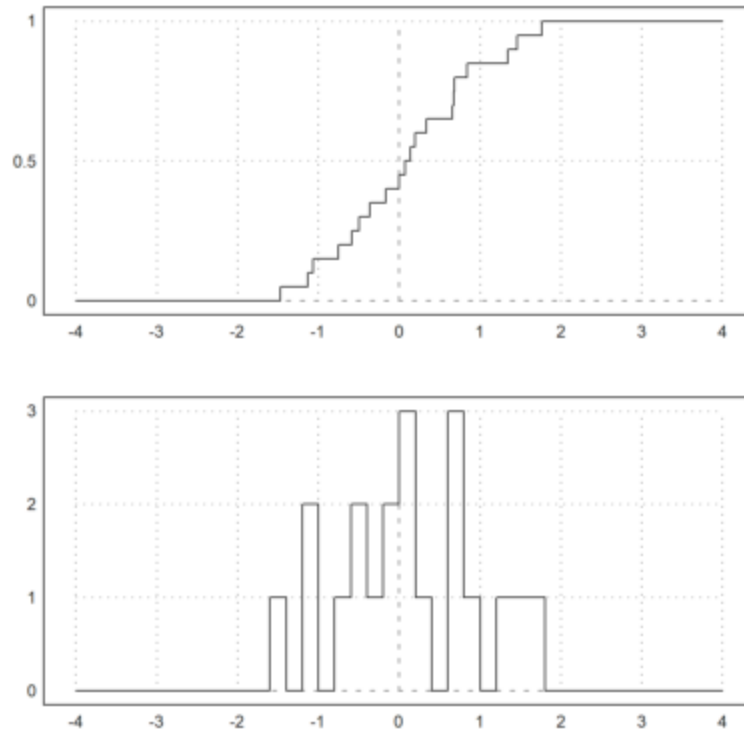
```
>x=normal(1,20);
```

Fungsi `empdist(x,vs)` membutuhkan array nilai yang diurutkan. Jadi kita harus mengurutkan `x` sebelum kita dapat menggunakannya.

```
>xs=sort(x);
```

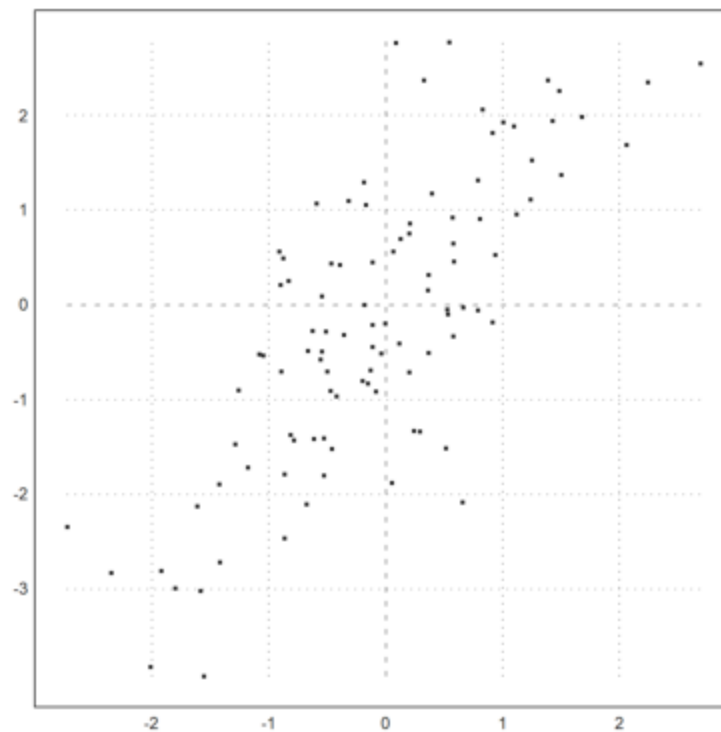
Kemudian kami memplot distribusi empiris dan beberapa batang kepadatan menjadi satu plot. Alih-alih plot batang untuk distribusi, kali ini kami menggunakan plot gigi gergaji.

```
>figure(2,1); ...
figure(1); plot2d("empdist",-4,4;xs); ...
figure(2); plot2d(histo(x,v=-4:0.2:4,<bar)); ...
figure(0):
```



Plot pencar mudah dilakukan di Euler dengan plot titik biasa. Grafik berikut menunjukkan bahwa X dan X+Y jelas berkorelasi positif.

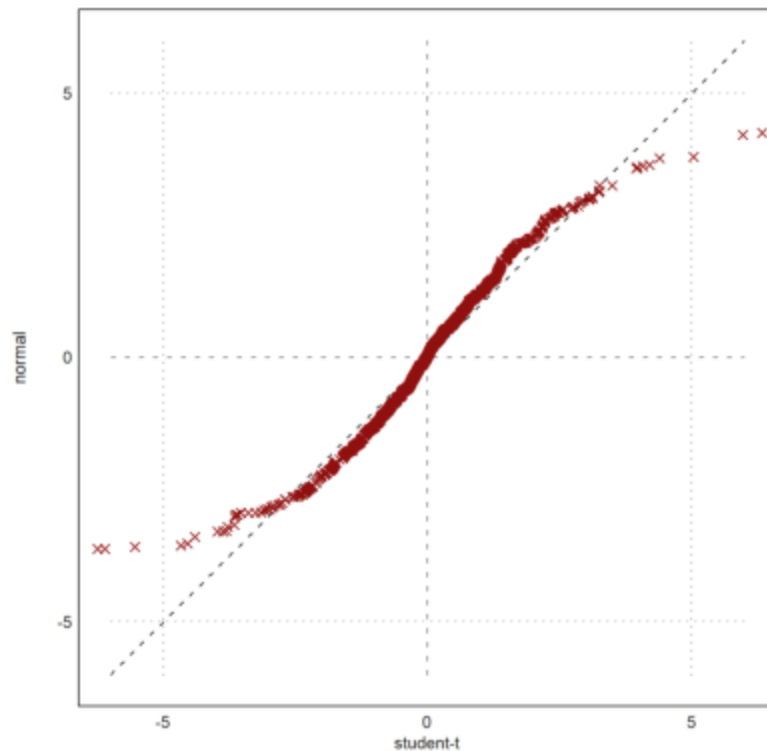
```
>x=normal(1,100); plot2d(x,x+rotright(x),>points,style="..") :
```



Seringkali, kami ingin membandingkan dua sampel dari distribusi yang berbeda. Ini dapat dilakukan dengan plot kuantil-kuantil.

Untuk pengujian, kami mencoba distribusi student-t dan distribusi eksponensial.

```
>x=randt(1,1000,5); y=randnormal(1,1000,mean(x),dev(x)); ...
plot2d("x",r=6,style="--",yl="normal",xl="student-t",>vertical); ...
plot2d(sort(x),sort(y),>points,color=red,style="x",>add):
```



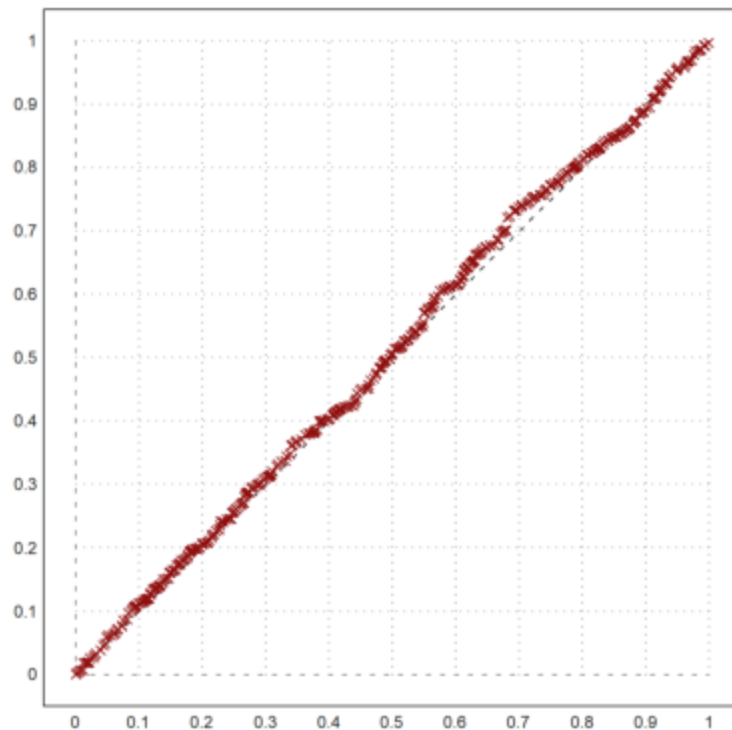
Plot jelas menunjukkan bahwa nilai terdistribusi normal cenderung lebih kecil di ujung ekstrim.

Jika kita memiliki dua distribusi dengan ukuran berbeda, kita dapat memperluas yang lebih kecil atau mengecilkan yang lebih besar. Fungsi berikut ini baik untuk keduanya. Dibutuhkan nilai median dengan persentase antara 0 dan 1.

```
>function medianexpand (x,n) := median(x,p=linspace(0,1,n-1));
```

Mari kita bandingkan dua distribusi yang sama.

```
>x=random(1000); y=random(400); ...
plot2d("x",0,1,style="--"); ...
plot2d(sort(medianexpand(x,400)),sort(y),>points,color=red,style="x",>add):
```



## Regresi dan Korelasi

Regresi linier dapat dilakukan dengan fungsi `polyfit()` atau berbagai fungsi fit.

Sebagai permulaan, kami menemukan garis regresi untuk data univariat dengan `polyfit(x,y,1)`.

```
>x=1:10; y=[2,3,1,5,6,3,7,8,9,8]; writetable(x'|y',labc=["x","y"])
```

x	y
1	2
2	3
3	1
4	5
5	6
6	3
7	7
8	8
9	9
10	8

Kami ingin membandingkan kecocokan yang tidak berbobot dan berbobot. Pertama koefisien fit linier.

```
>p=polyfit(x,y,1)
```

```
[0.733333, 0.812121]
```

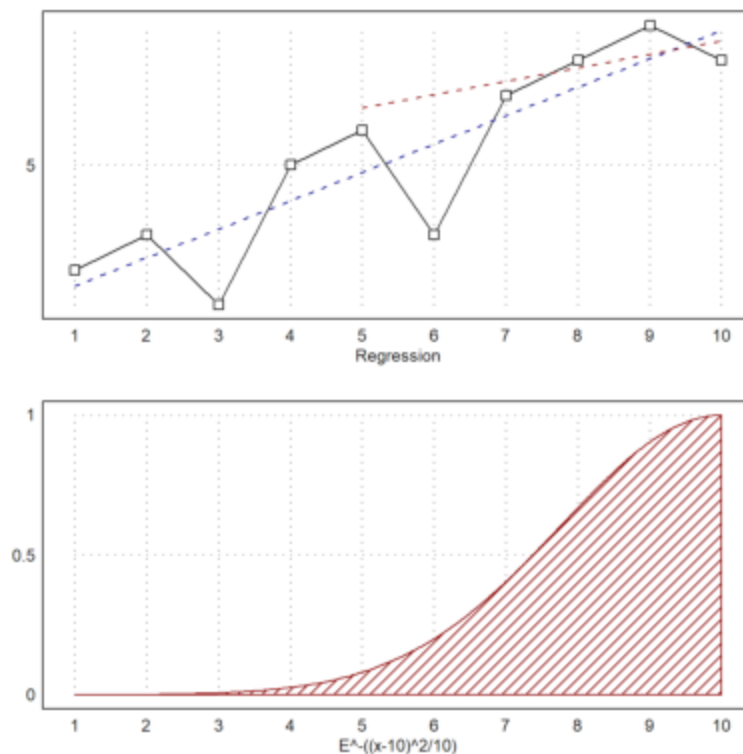
Sekarang koefisien dengan bobot yang menekankan nilai terakhir.

```
>w &= "exp(-(x-10)^2/10)"; pw=polyfit(x,y,1,w=w(x))
```

[4.71566, 0.38319]

Kami memasukkan semuanya ke dalam satu plot untuk poin dan garis regresi, dan untuk bobot yang digunakan.

```
>figure(2,1); ...
figure(1); statplot(x,y,"b",xl="Regression"); ...
plot2d("evalpoly(x,p)",>add,color=blue,style="--"); ...
plot2d("evalpoly(x,pw)",5,10,>add,color=red,style="--"); ...
figure(2); plot2d(w,1,10,>filled,style="/",fillcolor=red,xl=w); ...
figure(0):
```



Sebagai contoh lain kita membaca survei siswa, umur mereka, umur orang tua mereka dan jumlah saudara dari sebuah file.

Tabel ini berisi "m" dan "f" di kolom kedua. Kami menggunakan variabel tok2 untuk menyetel terjemahan yang tepat alih-alih membiarkan readtable() mengumpulkan terjemahan.

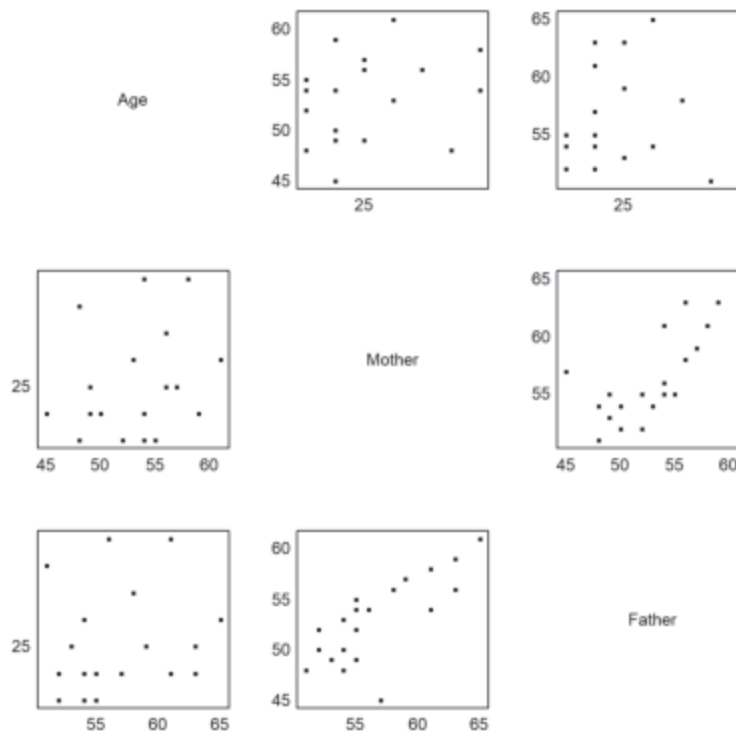
```
>{MS,hd}:=readtable("table1.dat",tok2:["m","f"]); ...
writetable(MS,labc=hd,tok2:["m","f"]);
```

Person	Sex	Age	Mother	Father	Siblings
1	m	29	58	61	1
2	f	26	53	54	2
3	m	24	49	55	1
4	f	25	56	63	3
5	f	25	49	53	0
6	f	23	55	55	2
7	m	23	48	54	2
8	m	27	56	58	1
9	m	25	57	59	1
10	m	24	50	54	1
11	f	26	61	65	1
12	m	24	50	52	1

13	m	29	54	56	1
14	m	28	48	51	2
15	f	23	52	52	1
16	m	24	45	57	1
17	f	24	59	63	0
18	f	23	52	55	1
19	m	24	54	61	2
20	f	23	54	55	1

Bagaimana usia bergantung satu sama lain? Kesan pertama berasal dari sebar berpasangan.

```
>scatterplots(tablecol(MS,3:5),hd[3:5]):
```



Jelas terlihat bahwa usia ayah dan ibu saling bergantung. Mari kita tentukan dan plot garis regresi.

```
>cs:=MS[,4:5]'; ps:=polyfit(cs[1],cs[2],1)
```

```
[17.3789, 0.740964]
```

Ini jelas model yang salah. Garis regresi adalah  $s=17+0,74t$ , di mana  $t$  adalah umur ibu dan  $s$  adalah umur ayah. Perbedaan usia mungkin sedikit bergantung pada usia, tetapi tidak terlalu banyak.

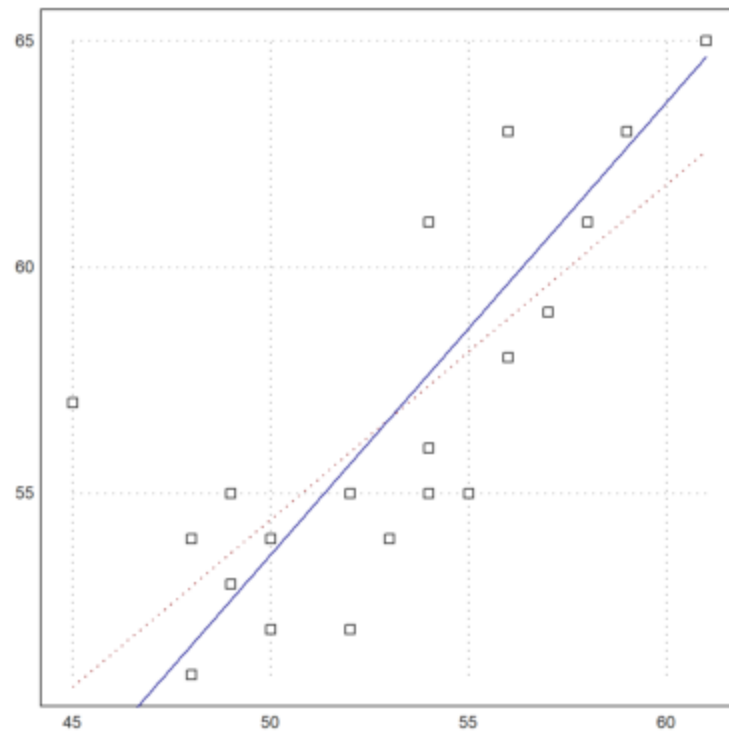
Sebaliknya, kami mencurigai fungsi seperti  $s=a+t$ . Maka  $a$  adalah rata-rata dari  $s-t$ . Ini adalah perbedaan usia rata-rata antara ayah dan ibu.

```
>da:=mean(cs[2]-cs[1])
```

```
3.65
```

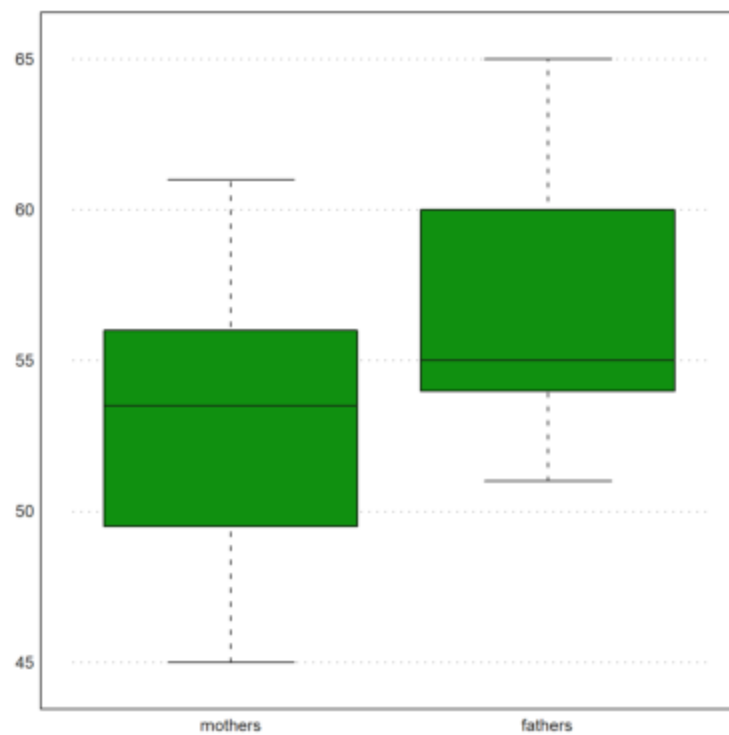
Mari kita plot ini menjadi satu plot pencar.

```
>plot2d(cs[1],cs[2],>points); ...
  plot2d("evalpoly(x,ps)",color=red,style=".",>add); ...
  plot2d("x+da",color=blue,>add):
```



Berikut adalah plot kotak dari dua zaman. Ini hanya menunjukkan, bahwa usianya berbeda.

```
>boxplot(cs,["mothers","fathers"]):
```



Menariknya, perbedaan median tidak sebesar perbedaan rata-rata.

```
>median(cs[2]) - median(cs[1])
```

```
1.5
```

Koefisien korelasi menunjukkan korelasi positif.

```
>correl(cs[1], cs[2])
```

```
0.7588307236
```

Korelasi peringkat adalah ukuran untuk urutan yang sama di kedua vektor. Ini juga cukup positif.

```
>rankcorrel(cs[1], cs[2])
```

```
0.758925292358
```

## Membuat Fungsi baru

Tentu saja, bahasa EMT dapat digunakan untuk memprogram fungsi-fungsi baru. Misalnya, kita mendefinisikan fungsi skewness.

lateks:  $\text{sk}(x) = \frac{\sqrt[n]{\sum_i (x_i - m)^3}}{\left(\sum_i (x_i - m)^2\right)^{3/2}}$

di mana  $m$  adalah rata-rata dari  $x$ .

```
>function skew (x:vector) ...
  m=mean(x);
  return sqrt(cols(x)) * sum((x-m)^3) / (sum((x-m)^2))^(3/2);
endfunction
```

Seperti yang Anda lihat, kita dapat dengan mudah menggunakan bahasa matriks untuk mendapatkan implementasi yang sangat singkat dan efisien. Mari kita coba fungsi ini.

```
>data=normal(20); skew(normal(10))
```

```
-0.198710316203
```

Ini adalah fungsi lain, yang disebut koefisien kemiringan Pearson.

```
>function skew1 (x) := 3 * (mean(x) - median(x)) / dev(x)
>skew1(data)
```

```
-0.0801873249135
```

## Simulasi Monte Carlo

Euler dapat digunakan untuk mensimulasikan kejadian acak. Kita telah melihat contoh sederhana di atas. Ini satu lagi, yang mensimulasikan 1000 kali 3 lemparan dadu, dan meminta distribusi jumlahnya.

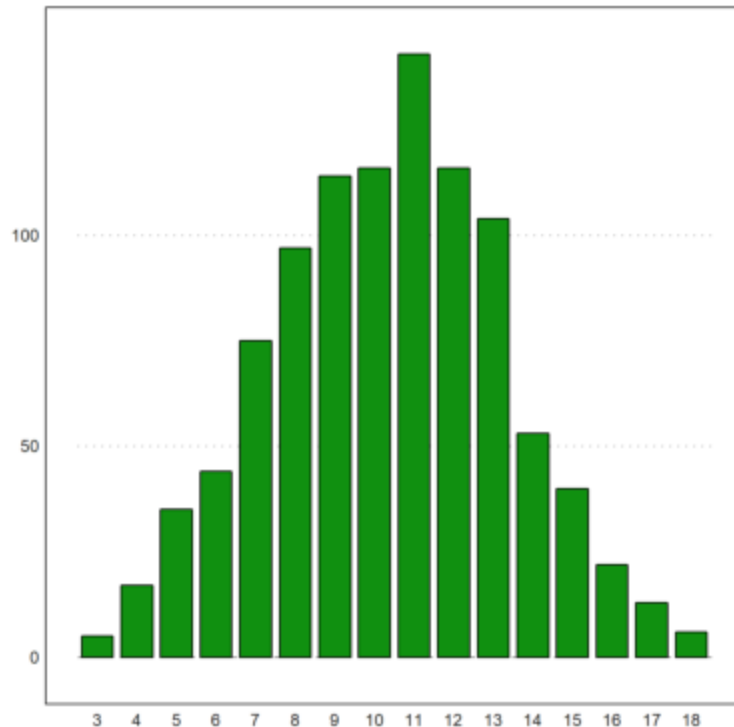


```
>ds:=sum(intrandom(1000,3,6)'); fs=getmultiplicities(3:18,ds)
```

```
[5, 17, 35, 44, 75, 97, 114, 116, 143, 116, 104, 53, 40,
22, 13, 6]
```

Kita bisa merencanakan ini sekarang.

```
>columnplot(fs,lab=3:18):
```



Untuk menentukan distribusi yang diharapkan tidak begitu mudah. Kami menggunakan rekursi lanjutan untuk ini.

Fungsi berikut menghitung banyaknya cara bilangan k dapat dinyatakan sebagai jumlah dari n bilangan dalam rentang 1 sampai m. Ini bekerja secara rekursif dengan cara yang jelas.

```
>function map countways (k; n, m) ...
  if n==1 then return k>=1 && k<=m
  else
    sum=0;
    loop 1 to m; sum=sum+countways(k-#,n-1,m); end;
    return sum;
  end;
endfunction
```

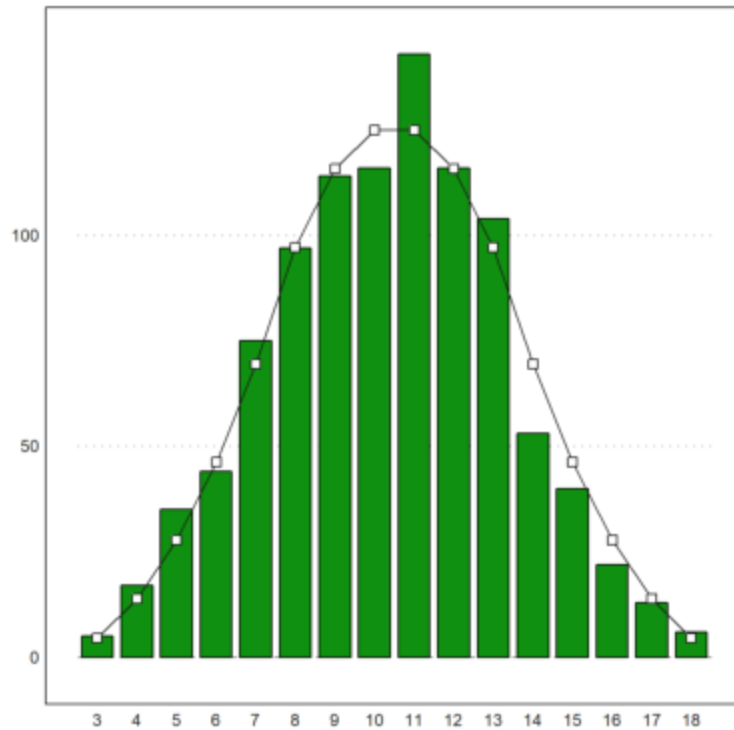
Inilah hasil lemparan dadu sebanyak tiga kali.

```
>cw=countways(3:18,3,6)
```

```
[1, 3, 6, 10, 15, 21, 25, 27, 27, 25, 21, 15, 10, 6, 3,
1]
```

Kami menambahkan nilai yang diharapkan ke plot.

```
>plot2d(cw/6^3*1000,>add); plot2d(cw/6^3*1000,>points,>add):
```



Untuk simulasi lain, penyimpangan nilai rata-rata  $n$  0-1-variabel acak terdistribusi normal adalah  $1/\sqrt{n}$ .

```
>longformat; 1/sqrt(10)
```

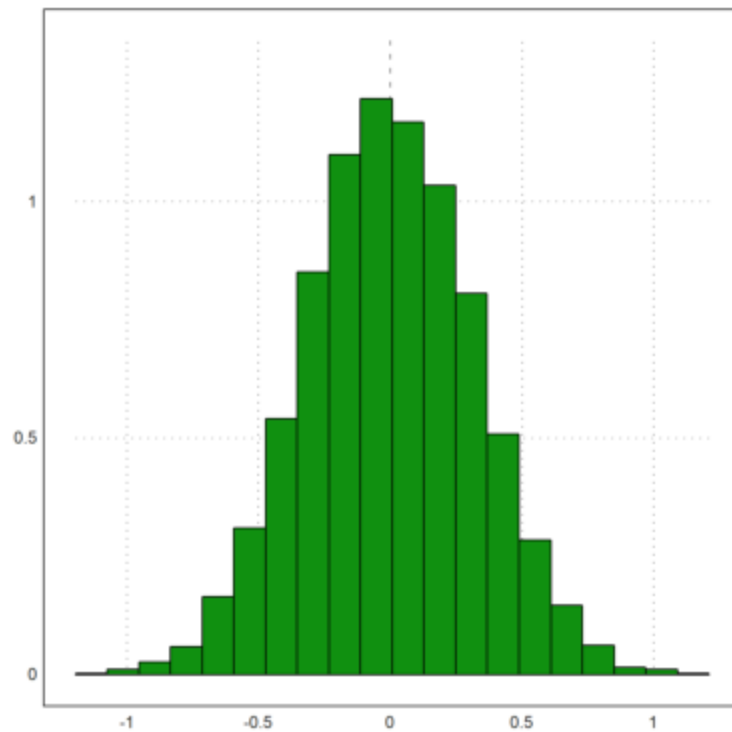
```
0.316227766017
```

Mari kita periksa ini dengan simulasi. Kami menghasilkan 10.000 kali 10 vektor acak.

```
>M=normal(10000,10); dev(mean(M)')
```

```
0.319493614817
```

```
>plot2d(mean(M) ',>distribution):
```



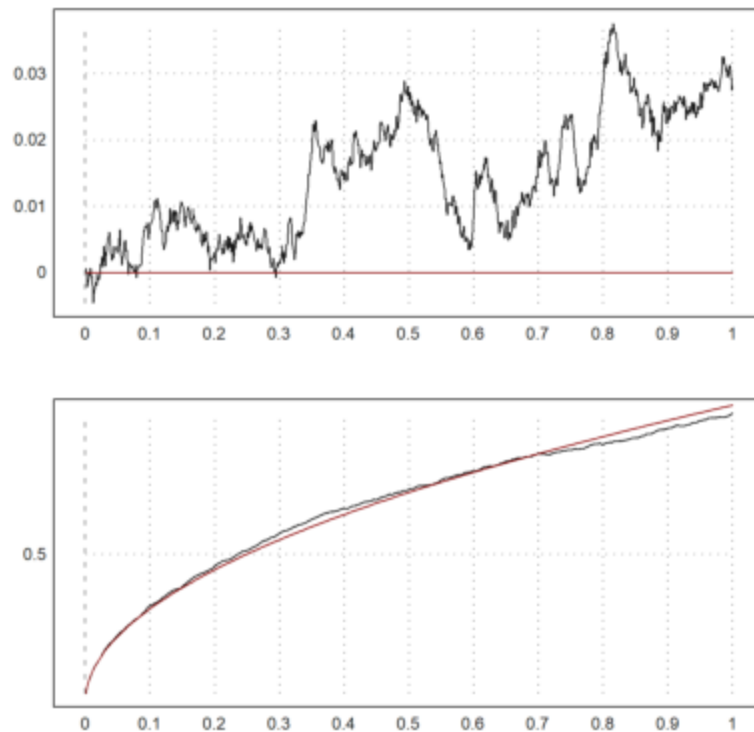
Median dari 10 bilangan acak terdistribusi 0-1-normal memiliki deviasi yang lebih besar.

```
>dev (median (M) ')
```

```
0.374460271535
```

Karena kita dapat dengan mudah membuat jalan acak, kita dapat mensimulasikan proses Wiener. Kami mengambil 1000 langkah dari 1000 proses. Kami kemudian memplot standar deviasi dan rata-rata langkah ke-n dari proses ini bersama dengan nilai yang diharapkan dalam warna merah.

```
>n=1000; m=1000; M=cumsum(normal(n,m)/sqrt(m)); ...
t=(1:n)/n; figure(2,1); ...
figure(1); plot2d(t,mean(M')'); plot2d(t,0,color=red,>add); ...
figure(2); plot2d(t,dev(M')'); plot2d(t,sqrt(t),color=red,>add); ...
figure(0):
```



## Tes

Tes adalah alat penting dalam statistik. Di Euler, banyak tes yang diterapkan. Semua tes ini mengembalikan kesalahan yang kami terima jika kami menolak hipotesis nol.

Sebagai contoh, kami menguji lemparan dadu untuk distribusi seragam. Pada 600 lemparan, kami mendapat nilai berikut, yang kami masukkan ke uji chi-square.

```
>chitest([90,103,114,101,103,89],dup(100,6)')
```

```
0.498830517952
```

Tes chi-kuadrat juga memiliki mode yang menggunakan simulasi Monte Carlo untuk menguji statistik. Hasilnya harus hampir sama. Parameter >p menginterpretasikan vektor-y sebagai vektor probabilitas.

```
>chitest([90,103,114,101,103,89],dup(1/6,6)',>p,>montecarlo)
```

```
0.526
```

Kesalahan ini terlalu besar. Jadi kita tidak bisa menolak pemerataan distribusi. Ini tidak membuktikan bahwa dadu kami adil. Tapi kita tidak bisa menolak hipotesis kita.

Selanjutnya kami menghasilkan 1000 lemparan dadu menggunakan generator angka acak, dan melakukan pengujian yang sama.

```
>n=1000; t=random([1,n*6]); chitest(count(t*6,6),dup(n,6)')
```

```
0.528028118442
```

Mari kita uji nilai rata-rata 100 dengan uji-t.

```
>s=200+normal([1,100])*10; ...  
ttest(mean(s),dev(s),100,200)
```

```
0.0218365848476
```

Fungsi `ttest()` membutuhkan nilai rata-rata, simpangan, jumlah data, dan nilai rata-rata untuk diuji.

Sekarang mari kita periksa dua pengukuran untuk rata-rata yang sama. Kami menolak hipotesis bahwa mereka memiliki rata-rata yang sama, jika hasilnya  $<0,05$ .

```
>tcomparedata(normal(1,10),normal(1,10))
```

```
0.38722000942
```

Jika kami menambahkan bias ke satu distribusi, kami mendapat lebih banyak penolakan. Ulangi simulasi ini beberapa kali untuk melihat efeknya.

```
>tcomparedata(normal(1,10),normal(1,10)+2)
```

```
5.60009101758e-07
```

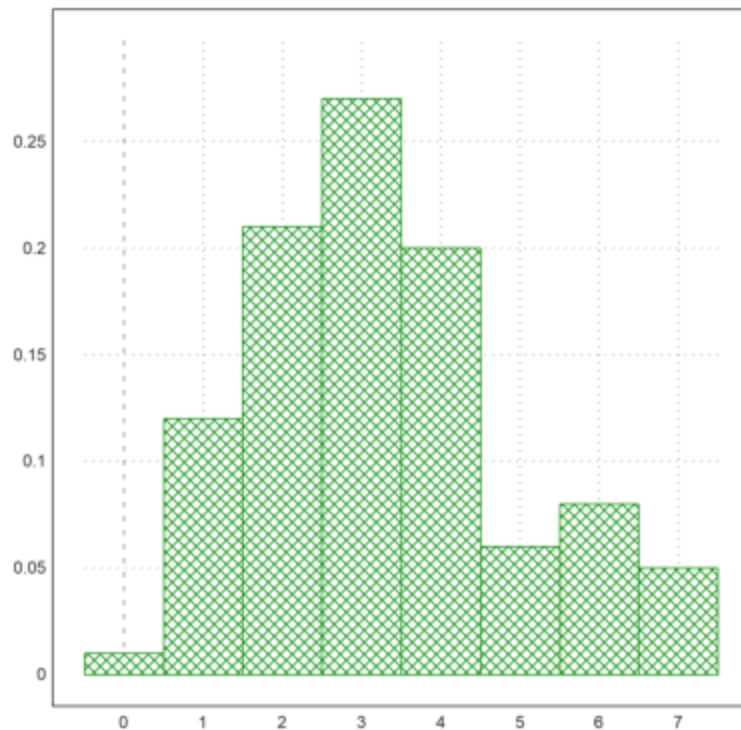
Dalam contoh berikutnya, kami menghasilkan 20 lemparan dadu acak 100 kali dan menghitungnya. Harus ada rata-rata  $20/6=3,3$ .

```
>R=random(100,20); R=sum(R*6<=1)'; mean(R)
```

```
3.28
```

Kami sekarang membandingkan jumlah satu dengan distribusi binomial. Pertama kita memplot distribusi satuan.

```
>plot2d(R,distribution=max(R)+1,even=1,style="\/"):
```



```
>t=count(R,21);
```

Kemudian kami menghitung nilai yang diharapkan.

```
>n=0:20; b=bin(20,n)*(1/6)^n*(5/6)^(20-n)*100;
```

Kita harus mengumpulkan beberapa angka untuk mendapatkan kategori yang cukup besar.

```
>t1=sum(t[1:2])|t[3:7]|sum(t[8:21]); ...
b1=sum(b[1:2])|b[3:7]|sum(b[8:21]);
```

Uji chi-square menolak hipotesis bahwa distribusi kita adalah distribusi binomial, jika hasilnya  $<0,05$ .

```
>chitest(t1,b1)
```

```
0.53921579764
```

Contoh berikut berisi hasil dari dua kelompok orang (pria dan wanita, katakanlah) memilih satu dari enam partai.

```
>A=[23,37,43,52,64,74;27,39,41,49,63,76]; ...
writetable(A,wc=6,labr=["m","f"],labc=1:6)
```

	1	2	3	4	5	6
m	23	37	43	52	64	74
f	27	39	41	49	63	76

Kami ingin menguji independensi suara dari jenis kelamin. Tes tabel  $\chi^2$  melakukan ini. Hasilnya terlalu besar untuk menolak kemerdekaan. Jadi kami tidak bisa mengatakan, jika pemungutan suara tergantung pada jenis kelamin dari data tersebut.

```
>tabletest (A)
```

```
0.990701632326
```

Berikut adalah tabel yang diharapkan, jika kita mengasumsikan frekuensi pemungutan suara yang diamati.

```
>writetable (expectedtable (A) ,wc=6,dc=1,labr=["m","f"],labc=1:6)
```

	1	2	3	4	5	6
m	24.9	37.9	41.9	50.3	63.3	74.7
f	25.1	38.1	42.1	50.7	63.7	75.3

Kita dapat menghitung koefisien kontingensi yang dikoreksi. Karena sangat mendekati 0, kami menyimpulkan bahwa pemungutan suara tidak bergantung pada jenis kelamin.

```
>contingency (A)
```

```
0.0427225484717
```

## Beberapa Tes Lagi

Selanjutnya kami menggunakan analisis varians (F-test) untuk menguji tiga sampel data yang terdistribusi normal untuk nilai rata-rata yang sama. Metode tersebut dinamakan ANOVA (analysis of variance). Di Euler, fungsi `varanalysis()` digunakan.

```
>x1=[109,111,98,119,91,118,109,99,115,109,94]; mean (x1) ,
```

```
106.545454545
```

```
>x2=[120,124,115,139,114,110,113,120,117]; mean (x2) ,
```

```
119.111111111
```

```
>x3=[120,112,115,110,105,134,105,130,121,111]; mean (x3)
```

```
116.3
```

```
>varanalysis (x1,x2,x3)
```

```
0.0138048221371
```

Ini berarti, kami menolak hipotesis dengan nilai rata-rata yang sama. Kami melakukan ini dengan probabilitas kesalahan 1,3%.

Ada juga uji median yang menolak sampel data dengan distribusi rata-rata yang berbeda menguji median sampel bersatu.

```
>a=[56,66,68,49,61,53,45,58,54];  
>b=[72,81,51,73,69,78,59,67,65,71,68,71];  
>mediantest(a,b)
```

```
0.0241724220052
```

Tes lain tentang kesetaraan adalah tes peringkat. Ini jauh lebih tajam daripada tes median.

```
>ranktest(a,b)
```

```
0.00199969612469
```

Dalam contoh berikut, kedua distribusi memiliki rata-rata yang sama.

```
>ranktest(random(1,100),random(1,50)*3-1)
```

```
0.129608141484
```

Mari kita coba mensimulasikan dua perlakuan a dan b yang diterapkan pada orang yang berbeda.

```
>a=[8.0,7.4,5.9,9.4,8.6,8.2,7.6,8.1,6.2,8.9];  
>b=[6.8,7.1,6.8,8.3,7.9,7.2,7.4,6.8,6.8,8.1];
```

Tes signum memutuskan, jika a lebih baik dari b.

```
>signtest(a,b)
```

```
0.0546875
```

Ini terlalu banyak kesalahan. Kita tidak dapat menolak bahwa a sama baiknya dengan b.

Tes Wilcoxon lebih tajam dari tes ini, tetapi bergantung pada nilai kuantitatif perbedaannya.

```
>wilcoxon(a,b)
```

```
0.0296680599405
```

Mari kita coba dua tes lagi menggunakan rangkaian yang dihasilkan.

```
>wilcoxon(normal(1,20),normal(1,20)-1)
```

```
0.0068706451766
```

```
>wilcoxon(normal(1,20),normal(1,20))
```

```
0.275145971064
```



Berikut ini adalah tes untuk generator angka acak. Euler menggunakan generator yang sangat bagus, jadi kita tidak perlu berharap ada masalah.

Pertama kami menghasilkan sepuluh juta angka acak di  $[0,1]$ .

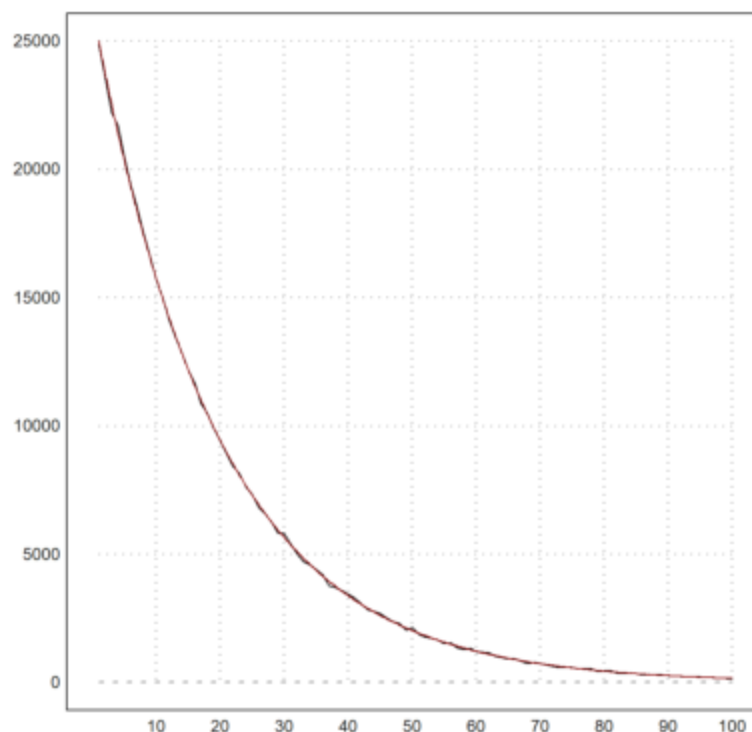
```
>n:=10000000; r:=random(1,n);
```

Selanjutnya kita menghitung jarak antara dua angka kurang dari 0,05.

```
>a:=0.05; d:=differences(nonzeros(r<a));
```

Akhirnya, kami memplot berapa kali, setiap jarak terjadi, dan membandingkannya dengan nilai yang diharapkan.

```
>m=getmultiplicities(1:100,d); plot2d(m); ...  
plot2d("n*(1-a)^(x-1)*a^2",color=red,>add):
```



Hapus datanya.

```
>remvalue n;
```

## Pengantar untuk Pengguna Proyek R

Jelas, EMT tidak bersaing dengan R sebagai paket statistik. Namun, ada banyak prosedur dan fungsi statistik yang tersedia di EMT juga. Jadi EMT dapat memenuhi kebutuhan dasar. Lagi pula, EMT hadir dengan paket numerik dan sistem aljabar komputer.

Notebook ini cocok untuk Anda jika sudah familiar dengan R, namun perlu mengetahui perbedaan sintaks EMT dan R. Kami mencoba memberikan gambaran umum tentang hal-hal yang jelas dan kurang jelas yang perlu Anda ketahui.

Selain itu, kami mencari cara untuk bertukar data antara kedua sistem.

Perhatikan bahwa ini adalah pekerjaan yang sedang berjalan.

## Sintaks Dasar

Hal pertama yang Anda pelajari di R adalah membuat vektor. Di EMT, perbedaan utamanya adalah operator : dapat mengambil ukuran langkah. Apalagi daya ikatnya rendah.

```
>n=10; 0:n/20:n-1
```

```
[0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5,
7, 7.5, 8, 8.5, 9]
```

Fungsi c() tidak ada. Dimungkinkan untuk menggunakan vektor untuk menggabungkan berbagai hal.

Contoh berikut, seperti banyak lainnya, dari "Introduction to R" yang disertakan dengan proyek R. Jika Anda membaca PDF ini, Anda akan menemukan bahwa saya mengikuti jalannya dalam tutorial ini.

```
>x=[10.4, 5.6, 3.1, 6.4, 21.7]; [x,0,x]
```

```
[10.4, 5.6, 3.1, 6.4, 21.7, 0, 10.4, 5.6, 3.1, 6.4, 21.7]
```

Operator titik dua dengan ukuran langkah EMT diganti dengan fungsi seq() di R. Kita bisa menulis fungsi ini di EMT.

```
>function seq(a,b,c) := a:b:c; ...
seq(0,-0.1,-1)
```

```
[0, -0.1, -0.2, -0.3, -0.4, -0.5, -0.6, -0.7, -0.8, -0.9, -1]
```

Fungsi rep() dari R tidak ada di EMT. Untuk input vektor, dapat ditulis sebagai berikut.

```
>function rep(x:vector,n:index) := flatten(dup(x,n)); ...
rep(x,2)
```

```
[10.4, 5.6, 3.1, 6.4, 21.7, 10.4, 5.6, 3.1, 6.4, 21.7]
```

Perhatikan bahwa "=" atau ":=" digunakan untuk tugas. Operator "->" digunakan untuk unit di EMT.

```
>125km -> " miles"
```

```
77.6713990297 miles
```

The "<-" operator for assignment is misleading anyway, and not a good idea of R. The following will compare a and -4 in EMT.

```
>a=2; a<-4
```

```
0
```

Di R, " $a < -4 < 3$ " berfungsi, tetapi " $a < -4 < -3$ " tidak. Saya juga memiliki ambiguitas serupa di EMT, tetapi mencoba menghilangkannya sedikit demi sedikit.

EMT dan R memiliki vektor tipe boolean. Namun dalam EMT, angka 0 dan 1 digunakan untuk mewakili salah dan benar. Di R, nilai benar dan salah tetap bisa digunakan dalam aritmatika biasa seperti di EMT.

```
>x<5, %*x
```

```
[0, 0, 1, 0, 0]
[0, 0, 3.1, 0, 0]
```

EMT melempar kesalahan atau menghasilkan NAN tergantung pada bendera "kesalahan".

```
>errors off; 0/0, isNAN(sqrt(-1)), errors on;
```

```
NAN
1
```

String sama di R dan EMT. Keduanya berada di lokal saat ini, bukan di Unicode.

Di R ada paket untuk Unicode. Di EMT, sebuah string dapat berupa string Unicode. String unicode dapat diterjemahkan ke pengkodean lokal dan sebaliknya. Selain itu, u"..." dapat berisi entitas HTML.

```
>u"&#169; Ren&eacute; Grothmann"
```

```
© René Grothmann
```

Berikut ini mungkin atau mungkin tidak ditampilkan dengan benar di sistem Anda sebagai A dengan titik dan garis di atasnya. Itu tergantung pada font yang Anda gunakan.

```
>chartoutf([480])
```

```
Ä
```

Penggabungan string dilakukan dengan "+" atau "|". Itu bisa termasuk angka, yang akan dicetak dalam format saat ini.

```
>"pi = "+pi
```

```
pi = 3.14159265359
```

## Pengindeksan

Sebagian besar waktu, ini akan berfungsi seperti di R.

Tetapi EMT akan menginterpretasikan indeks negatif dari belakang vektor, sedangkan R menginterpretasikan  $x[n]$  sebagai  $x$  tanpa elemen ke- $n$ .

```
>x, x[1:3], x[-2]
```

```
[10.4, 5.6, 3.1, 6.4, 21.7]
[10.4, 5.6, 3.1]
6.4
```

Perilaku R dapat dicapai dalam EMT dengan drop().

```
>drop(x,2)
```

```
[10.4, 3.1, 6.4, 21.7]
```

Vektor logis tidak diperlakukan berbeda sebagai indeks di EMT, berbeda dengan R. Anda perlu mengekstraksi elemen bukan nol terlebih dahulu di EMT.

```
>x, x>5, x[nonzeros(x>5)]
```

```
[10.4, 5.6, 3.1, 6.4, 21.7]
[1, 1, 0, 1, 1]
[10.4, 5.6, 6.4, 21.7]
```

Sama seperti di R, vektor indeks dapat berisi pengulangan.

```
>x[[1,2,2,1]]
```

```
[10.4, 5.6, 5.6, 10.4]
```

Tetapi nama untuk indeks tidak dimungkinkan di EMT. Untuk paket statistik, hal ini sering diperlukan untuk memudahkan akses ke elemen vektor.

Untuk meniru perilaku ini, kita dapat mendefinisikan fungsi sebagai berikut.

```
>function sel (v,i,s) := v[indexof(s,i)]; ...
s=["first","second","third","fourth"]; sel(x,["first","third"],s)
```

```
Trying to overwrite protected function sel!
Error in:
function sel (v,i,s) := v[indexof(s,i)]; ... ...
^
```

```
Trying to overwrite protected function sel!
Error in:
function sel (v,i,s) := v[indexof(s,i)]; ... ...
^
```

```
Trying to overwrite protected function sel!
Error in:
function sel (v,i,s) := v[indexof(s,i)]; ... ...
^
```

```
Trying to overwrite protected function sel!
Error in:
function sel (v,i,s) := v[indexof(s,i)]; ... ...
^
```

```
Trying to overwrite protected function sel!
Error in:
```

```
function sel (v,i,s) := v[indexof(s,i)]; ... ..
      ^
[10.4, 3.1]
```

## Tipe Data

EMT memiliki lebih banyak tipe data tetap daripada R. Jelas, di R terdapat vektor yang tumbuh. Anda dapat menyetel vektor numerik kosong  $v$  dan menetapkan nilai ke elemen  $v[17]$ . Ini tidak mungkin di EMT.

Berikut ini agak tidak efisien.

```
>v=[]; for i=1 to 10000; v=v|i; end;
```

EMT sekarang akan membuat vektor dengan  $v$  dan  $i$  ditambahkan pada tumpukan dan menyalin vektor itu kembali ke variabel global  $v$ .

Semakin efisien pra-mendefinisikan vektor.

```
>v=zeros(10000); for i=1 to 10000; v[i]=i; end;
```

Untuk mengubah jenis tanggal di EMT, Anda dapat menggunakan fungsi seperti `complex()`.

```
>complex(1:4)
```

```
[ 1+0i , 2+0i , 3+0i , 4+0i ]
```

Konversi ke string hanya dimungkinkan untuk tipe data dasar. Format saat ini digunakan untuk penggabungan string sederhana. Tapi ada fungsi seperti `print()` atau `frac()`.

Untuk vektor, Anda dapat dengan mudah menulis fungsi Anda sendiri.

```
>function tostr (v) ...
s="[";
loop 1 to length(v);
  s=s+print(v[#],2,0);
  if #<length(v) then s=s+","; endif;
end;
return s+"]";
endfunction
```

```
>tostr(linspace(0,1,10))
```

```
[0.00,0.10,0.20,0.30,0.40,0.50,0.60,0.70,0.80,0.90,1.00]
```

Untuk komunikasi dengan Maxima, terdapat fungsi `convertmxm()`, yang juga dapat digunakan untuk memformat vektor untuk output.

```
>convertmxm(1:10)
```

```
[1,2,3,4,5,6,7,8,9,10]
```

Untuk Lateks, perintah `tex` dapat digunakan untuk mendapatkan perintah Lateks.

```
>tex(&[1,2,3])
```

```
\left[ 1 , 2 , 3 \right]
```

## Faktor dan Tabel

Dalam pengantar R ada contoh dengan apa yang disebut faktor.

Berikut ini adalah daftar wilayah dari 30 negara bagian.

```
>austates = ["tas", "sa", "qld", "nsw", "nsw", "nt", "wa", "wa", ...
"qld", "vic", "nsw", "vic", "qld", "qld", "sa", "tas", ...
"sa", "nt", "wa", "vic", "qld", "nsw", "nsw", "wa", ...
"sa", "act", "nsw", "vic", "vic", "act"];
```

Asumsikan, kita memiliki pendapatan yang sesuai di setiap negara bagian.

```
>incomes = [60, 49, 40, 61, 64, 60, 59, 54, 62, 69, 70, 42, 56, ...
61, 61, 61, 58, 51, 48, 65, 49, 49, 41, 48, 52, 46, ...
59, 46, 58, 43];
```

Sekarang, kami ingin menghitung rata-rata pendapatan di wilayah tersebut. Menjadi program statistik, R memiliki `factor()` dan `tapply()` untuk ini.

EMT dapat melakukannya dengan menemukan indeks wilayah di daftar unik wilayah.

```
>auterr=sort(unique(austates)); f=indexofsorted(auterr,austates)

[6, 5, 4, 2, 2, 3, 8, 8, 4, 7, 2, 7, 4, 4, 5, 6, 5, 3,
8, 7, 4, 2, 2, 8, 5, 1, 2, 7, 7, 1]
```

Pada saat itu, kita dapat menulis fungsi loop kita sendiri untuk melakukan sesuatu hanya untuk satu faktor.

Atau kita bisa meniru fungsi `tapply()` dengan cara berikut.

```
>function map_tappl (i; f$:call, cat, x) ...
u=sort(unique(cat));
f=indexof(u,cat);
return f$(x[nonzeros(f==indexof(u,i))]);
endfunction
```

Ini sedikit tidak efisien, karena menghitung wilayah unik untuk setiap `i`, tetapi berhasil.

```
>tappl(auterr,"mean",austates,incomes)

[44.5, 57.333333333333, 55.5, 53.6, 55, 60.5, 56, 52.25]
```

Perhatikan bahwa ini berfungsi untuk setiap vektor wilayah.

```
>tappl(["act","nsw"],"mean",austates,incomes)

[44.5, 57.333333333333]
```

Sekarang, paket statistik EMT mendefinisikan tabel seperti pada R. Fungsi `readtable()` dan `writetable()` dapat digunakan untuk input dan output.

Sehingga kita bisa mencetak rata-rata pendapatan negara di daerah dengan cara yang bersahabat.

```
>writetable(tappl(auterr,"mean",austates,incomes),labc=auterr,wc=7)
```

```
      act    nsw     nt    qld     sa    tas     vic     wa
44.5  57.33  55.5  53.6    55  60.5    56  52.25
```

Kami juga dapat mencoba meniru perilaku R sepenuhnya.

Faktor jelas harus disimpan dalam kumpulan dengan jenis dan kategori (negara bagian dan teritori dalam contoh kita). Untuk EMT, kami menambahkan indeks yang telah dihitung sebelumnya.

```
>function makef (t) ...
## Factor data
## Returns a collection with data t, unique data, indices.
## See: tapply
u=sort(unique(t));
return {{t,u,indexofsorted(u,t)}};
endfunction
```

```
>statef=makef(austates);
```

Sekarang elemen ketiga dari koleksi akan berisi indeks.

```
>statef[3]
```

```
[6, 5, 4, 2, 2, 3, 8, 8, 4, 7, 2, 7, 4, 4, 5, 6, 5, 3,
8, 7, 4, 2, 2, 8, 5, 1, 2, 7, 7, 1]
```

Sekarang kita bisa meniru `tapply()` dengan cara berikut. Ini akan mengembalikan tabel sebagai kumpulan data tabel dan judul kolom.

```
>function tapply (t:vector,tf,f$:call) ...
## Makes a table of data and factors
## tf : output of makef()
## See: makef
uf=tf[2]; f=tf[3]; x=zeros(length(uf));
for i=1 to length(uf);
  ind=nonzeros(f==i);
  if length(ind)==0 then x[i]=NAN;
  else x[i]=f$(t[ind]);
endif;
end;
return {{x,uf}};
endfunction
```

Kami tidak menambahkan banyak pengecekan tipe di sini. Satu-satunya tindakan pencegahan menyangkut kategori (faktor) tanpa data. Tetapi orang harus memeriksa panjang `t` yang benar dan kebenaran koleksi `tf`.

Tabel ini dapat dicetak sebagai tabel dengan `writetable()`.

```
>writetable(tapply(incomes,statef,"mean"),wc=7)
```

act	nsw	nt	qld	sa	tas	vic	wa
44.5	57.33	55.5	53.6	55	60.5	56	52.25

## Array

EMT hanya memiliki dua dimensi untuk array. Tipe datanya disebut matriks. Namun, akan mudah untuk menulis fungsi untuk dimensi yang lebih tinggi atau pustaka C untuk ini.

R memiliki lebih dari dua dimensi. Di R array adalah vektor dengan bidang dimensi.

Dalam EMT, vektor adalah matriks dengan satu baris. Itu dapat dibuat menjadi matriks dengan `redim()`.

```
>shortformat; X=redim(1:20,4,5)
```

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

Ekstraksi baris dan kolom, atau sub-matriks, sangat mirip dengan R.

```
>X[,2:3]
```

2	3
7	8
12	13
17	18

Namun, dalam R dimungkinkan untuk menetapkan daftar indeks spesifik vektor ke suatu nilai. Hal yang sama dimungkinkan di EMT hanya dengan satu putaran.

```
>function setmatrixvalue (M, i, j, v) ...
  loop 1 to max(length(i),length(j),length(v))
    M[i{#},j{#}] = v{#};
  end;
endfunction
```

Kami mendemonstrasikan ini untuk menunjukkan bahwa matriks dilewatkan dengan referensi di EMT. Jika Anda tidak ingin mengubah matriks asli M, Anda perlu menyalinnya ke dalam fungsi.

```
>setmatrixvalue(X,1:3,3:-1:1,0); X,
```

1	2	0	4	5
6	0	8	9	10
0	12	13	14	15
16	17	18	19	20

Produk luar di EMT hanya dapat dilakukan di antara vektor. Ini otomatis karena bahasa matriks. Satu vektor harus berupa vektor kolom dan yang lainnya vektor baris.

```
>(1:5)*(1:5)'
```



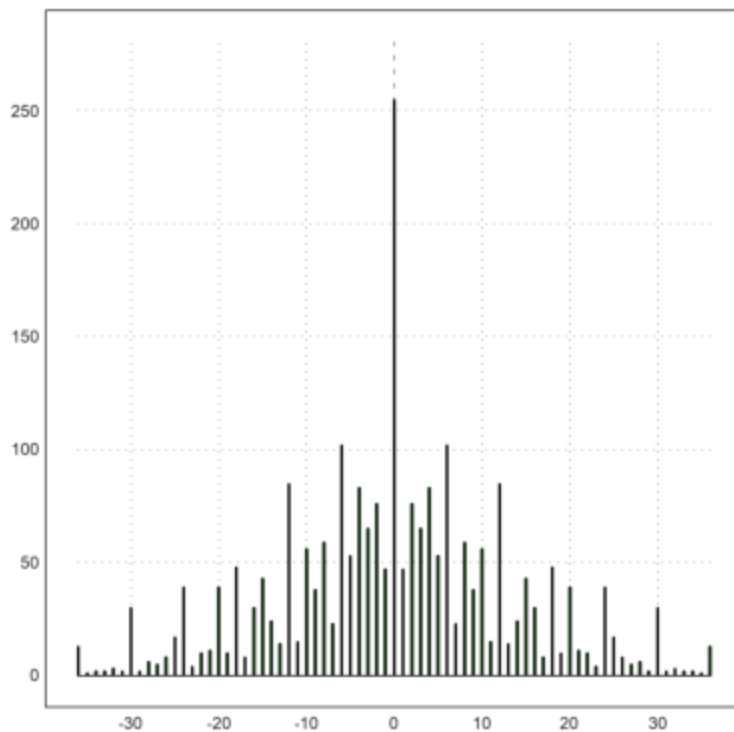
1	2	3	4	5
2	4	6	8	10
3	6	9	12	15
4	8	12	16	20
5	10	15	20	25

Dalam pengantar PDF untuk R ada contoh, yang menghitung distribusi ab-cd untuk a,b,c,d dipilih dari 0 sampai n secara acak. Solusi dalam R adalah membentuk matriks 4 dimensi dan menjalankan table() di atasnya.

Tentu saja, ini bisa dicapai dengan satu putaran. Tapi loop tidak efektif di EMT atau R. Di EMT, kita bisa menulis loop di C dan itu akan menjadi solusi tercepat.

Tapi kami ingin meniru perilaku R. Untuk ini, kami perlu meratakan perkalian ab dan membuat matriks ab-cd.

```
>a=0:6; b=a'; p=flatten(a*b); q=flatten(p-p'); ...
u=sort(unique(q)); f=getmultiplicities(u,q); ...
statplot(u,f,"h"):
```



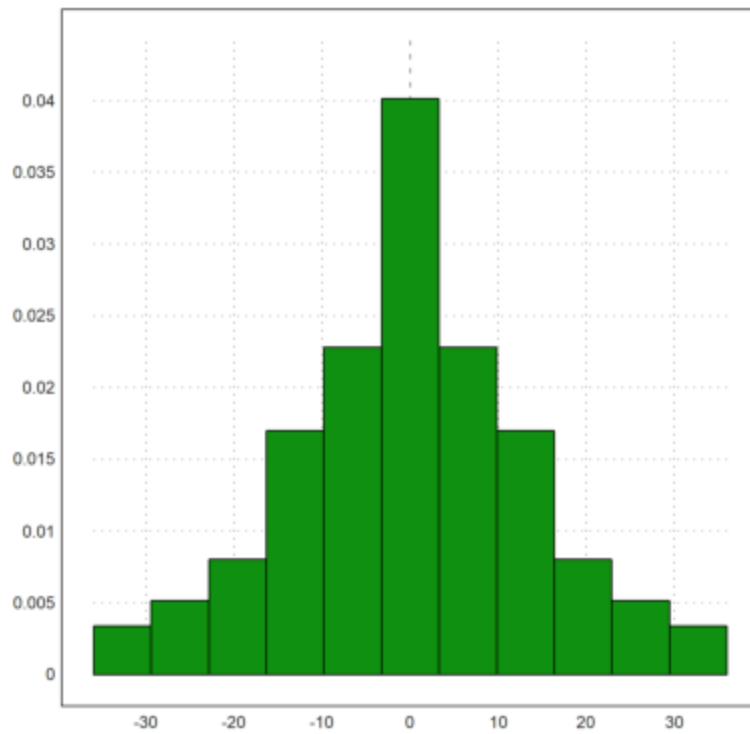
Selain perkalian yang tepat, EMT dapat menghitung frekuensi dalam vektor.

```
>getfrequencies(q,-50:10:50)
```

```
[0, 23, 132, 316, 602, 801, 333, 141, 53, 0]
```

Cara paling mudah untuk memplot ini sebagai distribusi adalah sebagai berikut.

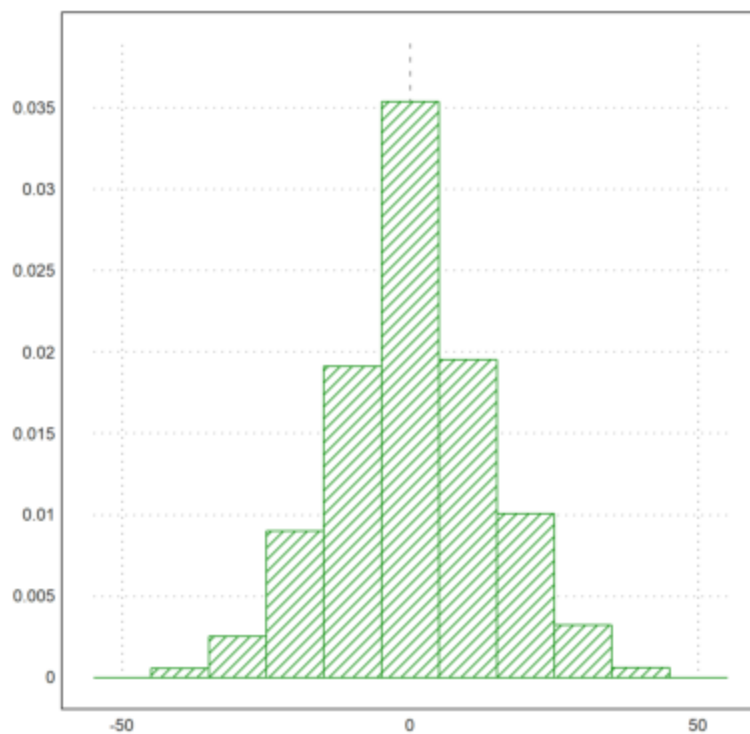
```
>plot2d(q,distribution=11):
```



Tetapi juga memungkinkan untuk melakukan pra-perhitungan hitungan dalam interval yang dipilih sebelumnya. Tentu saja, berikut ini menggunakan `getfrequencies()` secara internal.

Karena fungsi `histo()` mengembalikan frekuensi, kita perlu menskalakannya sehingga integral di bawah grafik batang adalah 1.

```
>{x,y}=histo(q,v=-55:10:55); y=y/sum(y)/differences(x); ...
plot2d(x,y,>bar,style="/"):
```



EMT memiliki dua jenis daftar. Salah satunya adalah daftar global yang bisa berubah, dan yang lainnya adalah tipe daftar yang tidak bisa diubah. Kami tidak peduli dengan daftar global di sini.

Jenis daftar yang tidak dapat diubah disebut koleksi di EMT. Ini berperilaku seperti struktur di C, tetapi elemennya hanya diberi nomor dan tidak diberi nama.

```
>L={{ "Fred", "Flintstone", 40, [1990, 1992] }}
```

```
Fred
Flintstone
40
[1990, 1992]
```

Saat ini elemen tidak memiliki nama, meskipun nama dapat diatur untuk tujuan khusus. Mereka diakses oleh nomor.

```
>(L[4]) [2]
```

```
1992
```

## File Input dan Output (Membaca dan Menulis Data)

Anda sering ingin mengimpor matriks data dari sumber lain ke EMT. Tutorial ini memberitahu Anda tentang banyak cara untuk mencapai hal ini. Fungsi sederhana adalah `writematrix()` dan `readmatrix()`.

Mari kita tunjukkan cara membaca dan menulis vektor real ke file.

```
>a=random(1,100); mean(a), dev(a),
```

```
0.49815
0.28037
```

Untuk menulis data ke file, kami menggunakan fungsi `writematrix()`.

Karena pengantar ini kemungkinan besar ada di direktori, di mana pengguna tidak memiliki akses tulis, kami menulis data ke direktori home pengguna. Untuk buku catatan sendiri, hal ini tidak diperlukan, karena file data akan ditulis ke dalam direktori yang sama.

```
>filename="test.dat";
```

Sekarang kita menulis vektor kolom `a'` ke file. Ini menghasilkan satu nomor di setiap baris file.

```
>writematrix(a', filename);
```

Untuk membaca data, kami menggunakan `readmatrix()`.

```
>a=readmatrix(filename)';
```

Dan hapus file tersebut.

```
>fileremove(filename);
>mean(a), dev(a),
```

```
0.49815
0.28037
```

Fungsi `writematrix()` atau `writetable()` dapat dikonfigurasi untuk bahasa lain.

Misalnya, jika Anda memiliki sistem bahasa Indonesia (titik desimal dengan koma), Excel Anda memerlukan nilai dengan koma desimal yang dipisahkan oleh titik koma dalam file csv (defaultnya adalah nilai yang dipisahkan koma). File berikut "test.csv" akan muncul di folder current Anda.

```
>filename="test.csv"; ...
writematrix(random(5,3),file=filename,separator=",");
```

Anda sekarang dapat membuka file ini dengan Excel bahasa Indonesia secara langsung.

```
>fileremove(filename);
```

Terkadang kami memiliki string dengan token seperti berikut ini.

```
>s1:"f m m f m m m f f f m m f"; ...
s2:"f f f m m f f";
```

Untuk menandai ini, kami mendefinisikan vektor token.

```
>tok:["f", "m"]
```

```
f
m
```

Kemudian kita dapat menghitung berapa kali setiap token muncul dalam string, dan memasukkan hasilnya ke dalam tabel.

```
>M:=getmultiplicities(tok, strtokens(s1))_ ...
getmultiplicities(tok, strtokens(s2));
```

Tulis tabel dengan header token.

```
>writetable(M, labc=tok, labr=1:2, wc=8)
```

	f	m
1	6	7
2	5	2

Untuk statika, EMT dapat membaca dan menulis tabel.

```
>file="test.dat"; open(file,"w"); ...
writeln("A,B,C"); writematrix(random(3,3)); ...
close();
```

The file looks like this.

```
>printfile(file)
```

```
A, B, C
0.7003664386138074, 0.1875530821001213, 0.3262339279660414
0.5926249243193858, 0.1522927283984059, 0.368140583062521
0.8065535209872989, 0.7265910840408142, 0.7332619844597152
```

Fungsi `readtable()` dalam bentuknya yang paling sederhana dapat membaca ini dan mengembalikan kumpulan nilai dan baris heading.

```
>L=readtable(file,>list);
```

Koleksi ini dapat dicetak dengan `writetable()` ke notebook, atau ke file.

```
>writetable(L,wc=10,dc=5)
```

A	B	C
0.70037	0.18755	0.32623
0.59262	0.15229	0.36814
0.80655	0.72659	0.73326

Matriks nilai adalah elemen pertama dari L. Perhatikan bahwa `mean()` dalam EMT menghitung nilai rata-rata dari baris matriks.

```
>mean(L[1])
```

```
0.40472
0.37102
0.75547
```

## File CSV

Pertama, mari kita menulis matriks ke dalam file. Untuk hasilnya, kami membuat file di direktori kerja saat ini.

```
>file="test.csv"; ...
M=random(3,3); writematrix(M,file);
```

Berikut adalah isi dari file ini.

```
>printfile(file)
```

```
0.8221197733097619, 0.821531098722547, 0.7771240608094004
0.8482947121863489, 0.3237767724883862, 0.6501422353377985
0.1482301827518109, 0.3297459716109594, 0.6261901074210923
```

CSV ini dapat dibuka pada sistem bahasa Inggris ke dalam Excel dengan klik dua kali. Jika Anda mendapatkan file seperti itu di sistem Jerman, Anda perlu mengimpor data ke Excel dengan memperhatikan titik desimal.

Tetapi titik desimal juga merupakan format default untuk EMT. Anda dapat membaca matriks dari file dengan `readmatrix()`.

```
>readmatrix(file)
```

```
0.82212 0.82153 0.77712
0.84829 0.32378 0.65014
0.14823 0.32975 0.62619
```

Dimungkinkan untuk menulis beberapa matriks ke satu file. Perintah `open()` dapat membuka file untuk ditulis dengan parameter "w". Standarnya adalah "r" untuk membaca.

```
>open(file,"w"); writematrix(M); writematrix(M'); close();
```

Matriks dipisahkan oleh garis kosong. Untuk membaca matriks, buka file dan panggil `readmatrix()` beberapa kali.

```
>open(file); A=readmatrix(); B=readmatrix(); A==B, close();
```

```
1      0      0
0      1      0
0      0      1
```

Di Excel atau spreadsheet serupa, Anda dapat mengekspor matriks sebagai CSV (nilai yang dipisahkan koma). Di Excel 2007, gunakan "simpan sebagai" dan "format lain", lalu pilih "CSV". Pastikan, tabel saat ini hanya berisi data yang ingin Anda ekspor.

Ini sebuah contoh.

```
>printfile("excel-data.csv")
```

```
0;1000;1000
1;1051,271096;1072,508181
2;1105,170918;1150,273799
3;1161,834243;1233,67806
4;1221,402758;1323,129812
5;1284,025417;1419,067549
6;1349,858808;1521,961556
7;1419,067549;1632,31622
8;1491,824698;1750,6725
9;1568,312185;1877,610579
10;1648,721271;2013,752707
```

Seperti yang Anda lihat, sistem Jerman saya menggunakan titik koma sebagai pemisah dan koma desimal. Anda dapat mengubahnya di pengaturan sistem atau di Excel, tetapi tidak perlu membaca matriks ke dalam EMT.

Cara termudah untuk membaca ini ke Euler adalah `readmatrix()`. Semua koma diganti dengan titik dengan parameter `>koma`. Untuk CSV bahasa Inggris, hilangkan saja parameter ini.

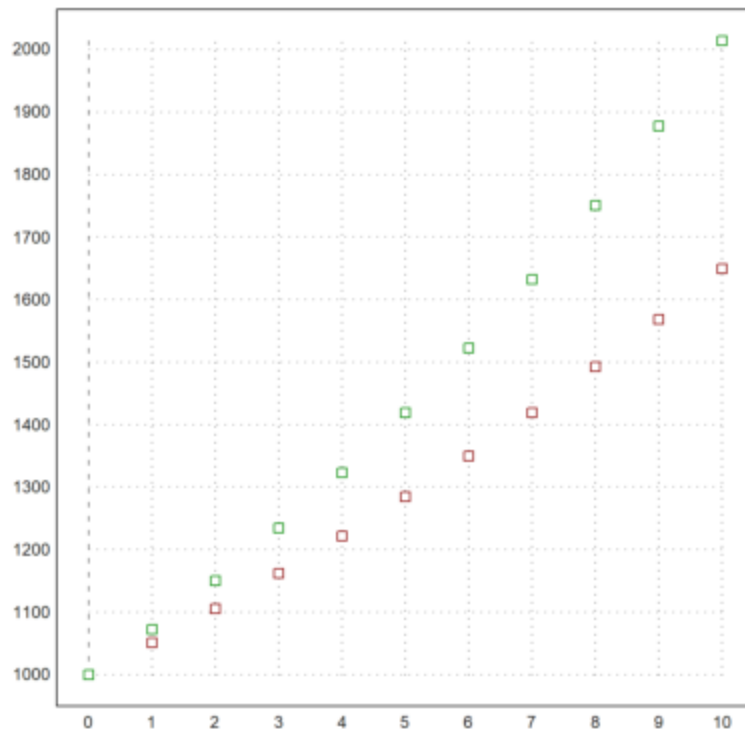
```
>M=readmatrix("excel-data.csv",>koma)
```

```
0      1000      1000
1      1051.3    1072.5
2      1105.2    1150.3
```

3	1161.8	1233.7
4	1221.4	1323.1
5	1284	1419.1
6	1349.9	1522
7	1419.1	1632.3
8	1491.8	1750.7
9	1568.3	1877.6
10	1648.7	2013.8

Let us plot this.

```
>plot2d(M'[1],M'[2:3],>points,color=[red,green]')
```



Ada cara yang lebih mendasar untuk membaca data dari file. Anda dapat membuka file dan membaca angka baris demi baris. Fungsi `getvectorline()` akan membaca angka dari baris data. Secara default, ini mengharapkan titik desimal. Tapi itu juga bisa menggunakan koma desimal, jika Anda memanggil `setdecimaldot(",")` sebelum Anda menggunakan fungsi ini.

Fungsi berikut adalah contoh untuk ini. Itu akan berhenti di akhir file atau baris kosong.

```
>function myload (file) ...
open(file);
M=[];
repeat
  until eof();
  v=getvectorline(3);
  if length(v)>0 then M=M_v; else break; endif;
end;
return M;
close(file);
endfunction
```

```
>myload(file)
```

```
0.82212 0.82153 0.77712
0.84829 0.32378 0.65014
0.14823 0.32975 0.62619
```

Dimungkinkan juga untuk membaca semua angka dalam file itu dengan `getvector()`.

```
>open(file); v=getvector(10000); close(); redim(v[1:9],3,3)
```

```
0.82212 0.82153 0.77712
0.84829 0.32378 0.65014
0.14823 0.32975 0.62619
```

Thus it is very easy to save a vector of values, one value in each line and read back this vector.

```
>v=random(1000); mean(v)
```

```
0.50303
```

```
>writematrix(v',file); mean(readmatrix(file)')
```

```
0.50303
```

## Menggunakan Tabel

Tabel dapat digunakan untuk membaca atau menulis data numerik. Sebagai contoh, kami menulis tabel dengan tajuk baris dan kolom ke file.

```
>file="test.tab"; M=random(3,3); ...
open(file,"w"); ...
writetable(M,separator=",",labc=["one","two","three"]); ...
close(); ...
printfile(file)
```

```
one,two,three
0.09, 0.39, 0.86
0.39, 0.86, 0.71
0.2, 0.02, 0.83
```

Ini dapat diimpor ke Excel.

Untuk membaca file di EMT, kami menggunakan `readtable()`.

```
>{M,headings}=readtable(file,>clabs); ...
writetable(M,labc=headings)
```

```
one two three
0.09 0.39 0.86
0.39 0.86 0.71
0.2 0.02 0.83
```

## Menganalisis Garis



Anda bahkan dapat mengevaluasi setiap baris dengan tangan. Misalkan, kita memiliki garis dengan format berikut.

```
>line="2020-11-03, Tue, 1'114.05"
```

```
2020-11-03, Tue, 1'114.05
```

Pertama kita dapat menandai garis.

```
>vt=strtokens(line)
```

```
2020-11-03
Tue
1'114.05
```

Kemudian kita dapat mengevaluasi setiap elemen garis menggunakan evaluasi yang sesuai.

```
>day(vt[1]), ...
indexof(["mon", "tue", "wed", "thu", "fri", "sat", "sun"], tolower(vt[2])), ...
strrepl(vt[3], "", "") ()
```

```
7.3816e+05
2
1114
```

Menggunakan ekspresi reguler, dimungkinkan untuk mengekstraksi hampir semua informasi dari sebaris data.

Asumsikan kita memiliki baris berikut sebuah dokumen HTML.

```
>line="<tr><td>1145.45</td><td>5.6</td><td>-4.5</td><tr>"
```

```
<tr><td>1145.45</td><td>5.6</td><td>-4.5</td><tr>
```

Untuk mengekstrak ini, kami menggunakan ekspresi reguler, yang mencari

- tanda kurung tutup >,
- string apa pun yang tidak mengandung tanda kurung dengan

sub-pertandingan "...",

- braket pembuka dan penutup menggunakan solusi terpendek,
- sekali lagi string apa pun yang tidak mengandung tanda kurung,
- dan tanda kurung buka <.

Ekspresi reguler agak sulit dipelajari tetapi sangat kuat.

```
>{pos, s, vt}=strxfind(line, ">([<>]+)<.+?>([<>]+<");
```

Hasilnya adalah posisi kecocokan, string yang cocok, dan vektor string untuk sub-kecocokan.

```
>for k=1:length(vt); vt[k](), end;
```

```
1145.5
5.6
```

Ini adalah fungsi yang membaca semua item numerik antara <td> dan </td>.

```
>function readtd (line) ...
v=[]; cp=0;
repeat
  {pos,s,vt}=strxfind(line,"<td.*?>(.*?)</td>",cp);
  until pos==0;
  if length(vt)>0 then v=v|vt[1]; endif;
  cp=pos+strlen(s);
end;
return v;
endfunction
```

```
>readtd(line+"<td>non-numerical</td>")
```

```
1145.45
5.6
-4.5
non-numerical
```

## Membaca dari Web

Situs web atau file dengan URL dapat dibuka di EMT dan dapat dibaca baris demi baris.

Dalam contoh, kami membaca versi terkini dari situs EMT. Kami menggunakan ekspresi reguler untuk memindai "Versi ..." dalam judul.

```
>function readversion () ...
urlopen("http://www.euler-math-toolbox.de/Programs/Changes.html");
repeat
  until urleof();
  s=urlgetline();
  k=strfind(s,"Version ",1);
  if k>0 then substring(s,k,strfind(s,"<",k)-1), break; endif;
end;
urlclose();
endfunction
```

```
>readversion
```

```
Version 2024-01-12
```

## Input dan Output Variabel

Anda dapat menulis variabel dalam bentuk definisi Euler ke file atau ke baris perintah.

```
>writevar(pi,"mypi");
```

```
mypi = 3.141592653589793;
```

Untuk pengujian, kami membuat file Euler di direktori kerja EMT.

```
>file="test.e"; ...
writevar(random(2,2),"M",file); ...
printf(file,3)

M = [ ..
0.5991820585590205, 0.7960280262224293;
0.5167243983231363, 0.2996684599070898];
```

We can now load the file. It will define the matrix M.

```
>load(file); show M,

M =
0.59918    0.79603
0.51672    0.29967
```

By the way, jika writevar() digunakan pada variabel, itu akan mencetak definisi variabel dengan nama variabel ini.

```
>writevar(M); writevar(inch$)

M = [ ..
0.5991820585590205, 0.7960280262224293;
0.5167243983231363, 0.2996684599070898];
inch$ = 0.0254;
```

Kami juga dapat membuka file baru atau menambahkan file yang sudah ada. Dalam contoh kami menambahkan file yang dihasilkan sebelumnya.

```
>open(file,"a"); ...
writevar(random(2,2),"M1"); ...
writevar(random(3,1),"M2"); ...
close();
>load(file); show M1; show M2;

M1 =
0.30287    0.15372
0.7504     0.75401
M2 =
0.27213
0.053211
0.70249
```

Untuk menghapus file apa pun gunakan fileremove().

```
>fileremove(file);
```

Vektor baris dalam file tidak memerlukan koma, jika setiap angka berada di baris baru. Mari kita buat file seperti itu, menulis setiap baris satu per satu dengan writeln().

```
>open(file,"w"); writeln("M = ["); ...
for i=1 to 5; writeln(" "+random()); end; ...
```

```
writeln("];"); close(); ...  
printfile(file)
```

```
M = [  
0.344851384551  
0.0807510017715  
0.876519562911  
0.754157709472  
0.688392638934  
];
```

```
>load(file); M
```

```
[0.34485, 0.080751, 0.87652, 0.75416, 0.68839]
```