Identitas Pribadi

Nama : Aura Yogi Azzahra
NIM : 23030630022
Kelas: Matematika B 2023

EMT untuk Statistika

Di 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 latar belakang untuk memahami detailnya.

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

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

```
  999
  999.9
  2.72641400622
```

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

```
>aspect(1.75); boxplot(M):
```

aspect (1.75) digunakan untuk mengatur rasio aspek dari plot (perbandingan antara lebar dan tinggi. boxplot(M) digunakan untuk mmebuat boxplot atau diagram kotak dari data di dalam variabel M. Boxplot adalah visualisasi statistik yang menunjukkan persebaran data, termasuk nilai minimum, median, dan nilai maksimum.

Contoh, kita asumsikan jumlah pria berikut dalam rentang ukuran tertentu.

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

Berikut adalah alur pendistribusiannya.

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

Kita bisa memasukkan data mentah tersebut ke dalam tabel.

Tabel adalah metode untuk menyimpan data statistik. Tabel kita harus berisi tiga kolom: Awal jangkauan, akhir jangkauan, jumlah pria 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=["BB","BA","Frek"])
```

BB	BA	Frek
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 memerlukan nilai rata-rata dan statistik ukuran lainnya, kita perlu menghitung titik tengah rentang tersebut. Kita bisa menggunakan dua kolom pertama tabel kita untuk ini.

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

```
>(T[,1]+T[,2])/2
```

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

```
>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 tertentu.

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

```
169.901234568
5.98912964449
```

Mari kita tambahkan distribusi nilai normal ke diagram batang di atas. Rumus distribusi normal dengan mean m dan simpangan baku d adalah:

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

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

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

Tabel

Di direktori buku catatan ini Anda menemukan file dengan tabel. Data tersebut merupakan hasil survei. Berikut adalah empat baris pertama file tersebut. Datanya 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 tokennya.

Untuk ini, kami mendefinisikan kumpulan token. Fungsi strtokens() mendapatkan vektor string token dari string tertentu.

```
>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 perlu menyediakannya dengan ":=".

```
>{MT,hd}=readtable("table.dat",tok2:=mf,tok4:=yn,tok5:=ev,tok7:=yn);
>load over statistics;
>writetable(MT[1:10],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
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

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

Jika kita tidak ingin menentukan token yang akan diterjemahkan 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);
```

ctok=[2,4,5,7]: Ini adalah untuk menentukan kolom yang akan diambil yaitu kolom ke-2, ke-4, ke-5, dan ke-7.

```
>tok
```

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

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

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

```
>MT[1]
```

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

Berikut isi tabel dengan nomor 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 keluaran readtable() ke dalam daftar.

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

Dengan menggunakan kolom token yang sama dan token yang dibaca dari file, kita dapat mencetak tabel. Kita dapat menentukan ctok, tok, dll. atau menggunakan tabel daftar.

```
>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, melewatkan baris apa pun dengan nilai NAN (”.” dalam file), dan indeks kolom, yang berisi nilai-nilai ini.

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

Kita bisa 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 suatu 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]
```

Kita bisa mencetak hasilnya di 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 v pada baris ke-5.

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

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

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

Person	Sex	Age	Titanic	Evaluation	Tip	Problem
2	f	23	y	g	1.8	n
3	f	26	y	g	1.8	y
6	m	28	y	g	2.8	y
18	m	38	y	g	.	n
16	m	26	y	g	2.8	n
15	f	31	y	g	0.8	n
12	m	32	y	g	1.8	n
23	f	38	y	g	2.8	n
14	f	25	y	g	1.8	y
9	f	24	y	vg	1.8	y
7	f	31	y	vg	2.8	n
20	f	28	y	vg	1.8	n
22	f	28	y	vg	1.8	y
13	m	29	y	vg	1.8	y
11	f	23	y	vg	1.8	y

Untuk statistik selanjutnya, kami ingin menghubungkan dua kolom tabel. Jadi kita ekstrak kolom 2 dan 4 dan urutkan tabelnya.

```
>i=sortedrows(MT,[2,4]);  ...
>  writetable(tablecol(MT[i],[2,4])',ctok=[1,2],tok=tok)
```

m	n
m	n
m	n
m	n
m	n
m	n
m	n
m	y
m	y
m	y
m	y
m	y
f	n
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y
f	y

Dengan getstatistics(), kita juga bisa menghubungkan jumlah dalam dua kolom tabel satu sama lain.

```
>MT24=tablecol(MT,[2,4]); ...
>{xu1,xu2,count}=getstatistics(MT24[1],MT24[2]); ...
>writetable(count,labr=tok[xu1],labc=tok[xu2])
```

	n	y
m	7	5
f	1	12

Sebuah 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 filenya.

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

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

```
>p=normal(1,1000);
>plot2d(p,distribution=20,style="\/");
>plot2d("qnormal(x,0,1)",add=1):
```

p=normal(1,1000); digunakan untuk menciptakan 1000 sampel acak yang terdistribusi normal dengan mean (rata-rata) 1 dan standar deviasi 1000.
plot2d(”qnormal(x,0,1)”,add=1);
digunakan untuk menambahkan plot dari distribusi normal standar (dengan mean 0 dan standar deviasi 1) ke grafik yang sama. Fungsi qnormal(x,0,1) mengacu pada distribusi kumulatif dari variabel acak normal standar. add=1 menunjukkan bahwa grafik ini harus ditambahkan ke grafik yang sudah ada, bukan dibuat baru.

Perlu diperhatikan perbedaan antara bar plot (sampel) dan kurva normal (distribusi sebenarnya). Masukkan kembali ketiga 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, serta outlier.

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

Untuk menghasilkan bilangan bulat acak, Euler memiliki intrandom. 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 plot hasilnya menggunakan kolomplot().

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

Meskipun inrandom(n,m,k) mengembalikan bilangan bulat yang terdistribusi secara seragam dari 1 hingga k, distribusi bilangan bulat lainnya dapat digunakan 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. Lihat referensinya.

Misalnya, kita mencoba distribusi eksponensial. Variabel acak kontinu X dikatakan berdistribusi eksponensial, jika PDF-nya diberikan oleh

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

with parameter

$$\lambda = \frac{1}{\mu}, \quad \mu \text{ is the mean, and denoted by } X \sim \text{Exponential}(\lambda).$$

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

Parameter pertama (1) adalah lambda, yang merupakan parameter distribusi eksponensial.
Parameter kedua (1000) menunjukkan jumlah angka acak yang dihasilkan.
Parameter ketiga (2) bisa menunjukkan dimensi atau bentuk output.

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):
```

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

Peluang berada di kawasan hijau adalah sebagai berikut.

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

```
  0.248662156979
```

Ini dapat dihitung secara numerik dengan integral berikut.

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

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

```
  0.248662156979
```

Mari kita bandingkan distribusi binomial dengan distribusi normal yang mean dan deviasinya sama. Fungsi invbindis() menyelesaikan interpolasi linier antara nilai integer.

```
>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. Dengan demikian kita mendapatkan plot 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 akurat untuk mengevaluasi distribusi. Mari kita periksa chidis() dengan integral.

Penamaannya mencoba untuk konsisten. Misalnya.,

- distribusi chi-kuadratnya adalah chidis(),
- fungsi kebalikannya adalah invchidis(),
- kepadatannya 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

Distribusi diskret adalah jenis distribusi probabilitas yang digunakan untuk variabel acak diskret, yaitu variabel yang hanya dapat memiliki nilai tertentu, biasanya dalam bentuk bilangan bulat.

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

Pertama kita atur fungsi distribusinya.

```
>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]
```

Perintah ini menggunakan operator | dan + untuk membuat nilai dalam variabel wd.

1:6 Ini menghasilkan vektor [1, 2, 3, 4, 5, 6].
(1:6) + [-0.01, 0.01, 0, 0, 0, 0]: Operasi ini menambahkan kedua vektor elemen per elemen.
Hasilnya:

$$[1-0.01, 2+0.01, 3, 4, 5, 6] = [0.99, 2.01, 3, 4, 5, 6]$$

[1-0.01,2+0.01,3,4,5,6]=[0.99,2.01,3,4,5,6]/6 Membagi setiap elemen hasil penjumlahan tadi dengan 6.
Hasilnya:

$$[\frac{0.99}{6}, \frac{2.01}{6}, \frac{3}{6}, \frac{4}{6}, \frac{5}{6}, \frac{6}{6}] = [0.165, 0.335, 0.5, 0.6667, 0.8333, 1]$$

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

Ini hampir merupakan distribusi yang seragam. Mari kita tentukan generator nomor acak untuk ini. Fungsi find(v,x) mencari nilai x pada vektor v. Fungsi ini juga berfungsi untuk vektor x.

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

Kesalahannya sangat halus sehingga kita hanya melihatnya dengan banyak iterasi.

Fungsi wrongdice mengembalikan sebuah matriks berukuran n x m, di mana setiap elemen dari matriks ini adalah indeks posisi dari elemen wd yang paling sesuai (atau mendekati) nilai acak dari random(n, m).

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

Hasil columnsplot akan menunjukkan frekuensi relatif dari setiap angka (1 hingga 6), yang memungkinkan Anda untuk melihat apakah distribusi itu merata atau tidak.

Berikut adalah fungsi sederhana untuk memeriksa keseragaman distribusi nilai 1...K dalam v. Kita menerima hasilnya, jika untuk semua frekuensi

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

Metode tersebut merupakan metode statistik untuk menguji keseragaman distribusi. Distribusi dianggap seragam jika frekuensi setiap nilai dalam v mendekati frekuensi ideal 1/K, dengan deviasi yang tidak melebihi batas toleransi.

```
>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 fungsinya menolak distribusi seragam.

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

```
  0
```

Dan ia menerima generator acak bawaan.

Manual:
- Asumsi dadu, maka peluang setiap sisi = 1/6
Dalam 1 juta lemparan maka

$$1000000 \times \frac{1}{6} \approx 166667$$

- Frekuensi setiap sisi fr. Proporsi tiap sisi = fr/n
Misalkan frekuensi munculnya angka adalah

$$[160000, 170000, 180000, 150000, 170000, 170000]$$

Maka proporsi setiap angka:

$$\frac{[160000, 170000, 180000, 150000, 170000, 170000]}{1000000}$$

$$[0.16, 0.17, 0.18, 0.15, 0.17, 0.17]$$

- Deviasi maksimum fn/n - 1/K

$$\frac{1}{K} = \frac{1}{6} = 0.1667$$

$$([0.16, 0.17, 0.18, 0.15, 0.17, 0.17] - 0.1667)$$

$$max(-0.0067, 0.0033, 0.0133, -0.0167, 0.0033, 0.0033) = 0.0133$$

- Bandingkan dengan batas toleransi.

$$Batas = \frac{delta}{\sqrt{n}} = \frac{1}{\sqrt{1000000}} = \frac{1}{1000} = 0.001$$

$$0.0133 > 0.001$$

Hasil 0 di sini mengindikasikan bahwa fungsi checkrandom telah menentukan bahwa distribusi tidak seragam.

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

```
  1
```

checkrandom mengembalikan 1 atau true yang berarti bahwa distribusi dari 1 juta bilangan acak rentang 1 sampai 6 dianggap cukup seragam dalam batas toleransi yang ditetapkan.

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

Misal kita akan menghitung probabilitas dari distribusi binomial di mana terdapat 1000 percobaan (misalnya, 1000 kali pelemparan koin), dengan probabilitas sukses pada setiap percobaan sebesar 0.4, dan kita ingin mengetahui probabilitas mendapatkan tepat 410 sukses.
Secara matematis, ini dihitung dengan rumus:

$$P(X \leq 410) = \binom{1000}{410} \cdot (0.4)^{410} \cdot (0.6)^{1000-410}$$

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

```
  0.751401349654
```

```
>bindis(4,10,0.6)
```

```
0.1662386176
```

Manual:
Secara matematis, ini dihitung dengan rumus:

$$P(X \leq 4) = \binom{10}{4} \cdot (0.6)^4 \cdot (0.4)^{10-4}$$

- Untuk k = 0

$$P(X = 0) = \binom{10}{0} \cdot (0.6)^0 \cdot (0.4)^{10} \approx 0.00010$$

- Untuk k = 1

$$P(X = 1) = \binom{10}{1} \cdot (0.6)^1 \cdot (0.4)^9 \approx 0.00157$$

- Untuk k = 2

$$P(X = 2) = \binom{10}{2} \cdot (0.6)^2 \cdot (0.4)^8 \approx 0.01061$$

- Untuk k = 3

$$P(X = 3) = \binom{10}{3} \cdot (0.6)^3 \cdot (0.4)^7 \approx 0.04246$$

- Untuk k = 4

$$P(X = 4) = \binom{10}{4} \cdot (0.6)^4 \cdot (0.4)^6 \approx 0.11147$$

Maka,

$$P(X \leq 4) = P(X = 0) + P(X = 1) + P(X = 2) + P(X = 3) + P(X = 4)$$

$$P(X \leq 4) = 0.00010 + 0.00157 + 0.01061 + 0.04246 + 0.11147$$

$$P(X \leq 4) \approx 0.1662$$

Fungsi Beta terbalik digunakan untuk menghitung interval kepercayaan Clopper-Pearson untuk parameter p. Tingkat defaultnya adalah alfa.

Arti dari 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. Namun untuk n yang 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) di dua tangan (26 kartu). Mari kita hitung probabilitas distribusi yang lebih buruk dari 3:2 (misalnya 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

Merencanakan Data/ Plot Data

Untuk memetakan data, kami mencoba hasil pemilu Jerman sejak tahun 1990, 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 beberapa bagian, kami menggunakan rangkaian nama.

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

Mari kita cetak persentasenya dengan baik.

Pertama kita mengekstrak kolom yang diperlukan. Kolom 3 sampai 7 adalah kursi masing-masing partai, dan kolom 2 adalah jumlah kursi seluruhnya. Kolom 1 adalah tahun pemilihan.

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

Kemudian statistiknya kita cetak dalam bentuk tabel. Kami menggunakan nama sebagai header kolom, dan tahun sebagai header untuk baris. Lebar default untuk kolom adalah wc=10, tetapi kami lebih memilih keluaran 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 ini menjumlahkan persentase dua partai besar yang menunjukkan bahwa partai-partai kecil berhasil memperoleh suara 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 setiap pesta.

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

Sekarang kita bisa memplot hasil pemilu 2009 dan perubahannya menjadi satu plot dengan 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 memperlihatkan baris data statistik dalam bentuk kolom. Kami memberikan label untuk baris dan kolom. sudut adalah sudut pandang.

```
>columnsplot3d(BT,scols=P,srows=YT, ...
>  angle=30°,ccols=CP):
```

Representasi lainnya adalah plot mosaik. 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 elemen-elemennya.

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

Ini adalah 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):
```

Namun 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 memerlukan vektor nilai. Selain itu, ia dapat mengatur labelnya ke apa pun yang kita inginkan, kami telah mendemonstrasikannya di tutorial ini.

Ini adalah aplikasi lain, di mana kita menghitung karakter dalam sebuah kalimat dan membuat 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; ...
>columnsplot(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 cara memplot frekuensi bilangan dalam suatu vektor.
Kami membuat vektor bilangan acak bilangan bulat 1 hingga 6.

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

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

Kemudian ekstrak nomor 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 mendemonstrasikan fungsi distribusi nilai empiris.

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

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

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

Kemudian kita plot distribusi empiris dan beberapa batang kepadatan ke dalam 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 sebar 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 kita ingin membandingkan dua sampel dengan distribusi yang berbeda. Hal ini dapat dilakukan dengan plot kuantil-kuantil.

Untuk pengujiannya, 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 tersebut dengan jelas menunjukkan bahwa nilai terdistribusi normal cenderung lebih kecil di ujung ekstrim.

Jika kita mempunyai dua distribusi yang ukurannya berbeda, kita dapat memperluas distribusi yang lebih kecil atau mengecilkan distribusi 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 linier dapat dilakukan dengan fungsi polyfit() atau berbagai fungsi fit.

Sebagai permulaan kita 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 kecocokan linier.

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

```
[0.733333,  0.812121]
```

Regresi linear dapat ditulis dalam bentuk:

$$y = mx + b$$

dengan

$$m = \frac{n\sum xy - (\sum x)(\sum y)}{n\sum x^2 - (\sum x)^2}$$
$$b = \frac{\sum y - m(\sum x)}{n}$$

Kita hitung:

$$n = 10, x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], y = [2, 3, 1, 5, 6, 3, 7, 8, 9, 8]$$
$$\sum x = 55$$
$$\sum y = 52$$
$$\sum xy = 353$$
$$\sum x^2 = 385$$

Maka:

$$m = \frac{10(353) - (55)(52)}{10(385) - (55)^2} = \frac{3530 - 2860}{3850 - 3025} = 0.812121$$
$$b = \frac{52 - 0.812121(55)}{10} = 0.733333$$

Jadi, b, m = 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 titik 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):
```

Contoh lain kita membaca survei siswa, usia mereka, usia orang tua mereka dan jumlah saudara kandung dari sebuah file.

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

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

```
Could not open the file
table1.dat
for reading!
Try "trace errors" to inspect local variables after errors.
readtable:
    if filename!=none then open(filename,"r"); endif;
```

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

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

```
Variable or function MS not found.
Error in:
scatterplots(tablecol(MS,3:5),hd[3:5]): ...
                        ^
```

Jelas terlihat bahwa usia ayah dan ibu saling bergantung satu sama lain. Mari kita tentukan dan plot garis regresinya.

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

```
MS is not a variable!
Error in:
cs:=MS[,4:5]'; ps:=polyfit(cs[1],cs[2],1) ...
           ^
```

Ini jelas merupakan model yang salah. Garis regresinya adalah s=17+0,74t, dengan t adalah umur ibu dan s adalah umur ayah. Perbedaan usia mungkin sedikit bergantung pada usia, tapi tidak terlalu banyak.

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

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

```
  cs is not a variable!
  Error in:
  da:=mean(cs[2]-cs[1]) ...
               ^
```

Mari kita plot ini menjadi satu plot sebar.

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

```
  cs is not a variable!
  Error in:
  plot2d(cs[1],cs[2],>points);  plot2d("evalpoly(x,ps)",color=re ...
              ^
```

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

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

```
  Variable or function cs not found.
  Error in:
  boxplot(cs,["mothers","fathers"]): ...
            ^
```

Menariknya, perbedaan median tidak sebesar perbedaan mean.

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

```
  cs is not a variable!
  Error in:
  median(cs[2])-median(cs[1]) ...
              ^
```

Koefisien korelasi menunjukkan korelasi positif.

```
>Koefisien korelasi menunjukkan korelasi positif.correl(cs[1],cs[2])
```

```
  Variable Koefisien not found!
  Error in:
  Koefisien korelasi menunjukkan korelasi positif.correl(cs[1],c ...
            ^
```

Korelasi pangkat merupakan ukuran keteraturan yang sama pada kedua vektor. Hal ini juga cukup positif.

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

```
  cs is not a variable!
  Error in:
  rankcorrel(cs[1],cs[2]) ...
                   ^
```

Membuat Fungsi baru

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

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

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
```

Berikut adalah fungsi lainnya, yang disebut koefisien skewness Pearson.

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

```
  -0.0801873249135
```

Kita simulasikan variabel acak berdistribusi normal 1000-5 sebanyak sejuta kali. Untuk ini, kita gunakan fungsi normal(m,n), yang menghasilkan matriks nilai berdistribusi 0-1, atau normal(n) yang secara default bernilai m=1.

```
>n=1000000; x=normal(n)
```

```
[-1.9943,  1.1795,  1.18591,  -0.441316,  -0.488192,  -0.675366,
-1.58021,  0.962882,  -1.66963,  -0.819503,  -0.929235,  0.539573,
-0.606864,  -0.56878,  -0.087391,  -0.404174,  0.0788819,  1.04282,
0.876296,  -0.877162,  0.689615,  1.66396,  1.10287,  1.81334,
-0.679579,  -0.739293,  -0.0919214,  -0.0516635,  1.68059,  -0.8748,
-0.661822,  1.14053,  -0.0725198,  -0.653134,  0.338853,  0.158539,
-1.33566,  -1.21098,  0.252185,  0.750436,  -0.335005,  0.79995,
0.237049,  0.10056,  -0.909771,  -0.896389,  0.807295,  0.420089,
-2.60022,  -0.334883,  0.563393,  -1.51976,  -0.618486,  -0.757591,
-1.5655,  -1.14804,  -1.62101,  -0.767098,  1.41355,  0.28399,
1.31699,  0.325665,  0.123923,  -0.301708,  0.224152,  1.61033,
-0.928959,  0.112323,  2.06755,  -0.415012,  1.227,  -0.392912,
-0.82682,  -1.37726,  1.09679,  -2.83431,  0.310441,  1.333,
-0.774371,  1.51449,  0.0260218,  0.244161,  -1.25954,  0.833535,
-1.05672,  -3.99147,  -1.70256,  0.620494,  -0.98936,  -0.868208,
0.344441,  0.926404,  0.00735312,  -1.07317,  0.26722,  0.469612,
-1.44652,  -0.0850004,  -1.22639,  -0.120585,  -0.498201,  0.793201,
-2.71956,  -0.832553,  -1.50601,  -0.473103,  -0.857245,  -0.397193,
-0.368307,  -1.08578,  -1.60682,  -0.525828,  0.00660302,  -2.44655,
0.761081,  -0.0405162,  -0.440874,  -0.733893,  0.275494,  1.14203,
 ... ]
```

```
>n=1000000; x=normal(n)*5+1000
```

```
[1002.99,  1003.08,  996.596,  990.94,  998.608,  1005.05,  997.111,
1006.2,  1003.62,  988.073,  999.093,  1001.26,  999.921,  994.418,
995.893,  1008.1,  1000.85,  997.523,  997.748,  1000.46,  988.609,
1003.88,  996.944,  994.016,  1008.61,  997.575,  992.025,  993.416,
1003.18,  1003.39,  1001.73,  1005.57,  995.544,  1004.48,  996.802,
1004.15,  1005.39,  997.653,  1002.83,  1005.25,  998.235,  994.648,
998.584,  1000.88,  998.811,  994.607,  995.101,  995.2,  1002.16,
1009.75,  1000.77,  999.994,  1011.51,  1006.36,  990.598,  1007.2,
1000.18,  999.05,  1002.22,  1000.08,  993.234,  1000.81,  1004.78,
1002.42,  993.949,  1002.01,  995.443,  994.59,  1008.39,  1000.62,
995.42,  1000.23,  998.282,  998.84,  992.97,  1000.81,  993.34,
1003.17,  1001.89,  999.305,  1000.63,  992.78,  989.389,  998.162,
1001.79,  1008.18,  1003.34,  1001.35,  1001.08,  998.659,  998.586,
1000.31,  994.955,  994.737,  988.751,  991.572,  995.534,  999.851,
1004.17,  997.659,  993.513,  996.48,  999.944,  996.108,  998.717,
1002.66,  1001.39,  1003.17,  992.563,  998.983,  994.664,  997.407,
1001.87,  999.54,  993.469,  999.279,  1000.47,  996.123,  1000.8,
1003.22,  993.011,  998.241,  1000.07,  1001.6,  1002.9,  1004.22,
1002.22,  1005.54,  999.447,  1002.94,  1003.54,  999.995,  996.819,
1001.89,  1003.54,  1005.62,  1002.17,  998.897,  1005.83,  1006.59,
 ... ]
```

terdapat juga fungsi randnormal(n,m,mean,dev), yang dapat kita gunakan. Fungsi ini mematuhi skema penamaan ”rand...” untuk generator acak.

```
>n=1000000; x=randnormal(1,n,1000,5)
```

```
[1001.46,  1002.83,  995.424,  1010.42,  993.922,  995.65,  1002.73,
996.528,  998.885,  999.782,  999.165,  996.189,  1007.19,  1002.5,
1000.04,  1001.11,  995.786,  1000.87,  1007.36,  997.776,  989.569,
1000.06,  999.158,  996.89,  1000.07,  1000.75,  1003.66,  995.875,
1003.36,  992.127,  1003.03,  1006.05,  999.733,  999.09,  1005.97,
1003.15,  998.16,  989.205,  1002.45,  996.641,  1008.69,  1006.67,
992.93,  1003.96,  1002.56,  997.29,  995.598,  995.616,  1000.14,
1002.68,  984.552,  1004.04,  1008.83,  994.196,  991.508,  1003.67,
1000.44,  1001.38,  996.971,  1001.32,  998.596,  995.479,  1002.12,
990.516,  997.716,  1008.15,  1000.37,  1001.07,  996.014,  1001.07,
1003.76,  996.41,  992.867,  998.441,  993.741,  997.669,  1007.58,
1005.19,  997.211,  1000.75,  994.023,  994.398,  998.096,  994.683,
994.84,  1001.44,  1003.19,  987.72,  1003.07,  1008.01,  1009.52,
1007.39,  997.166,  1007.7,  987.807,  1002.73,  1003.18,  1002.86,
1002.75,  1001.91,  995.783,  998.36,  1002.57,  994.433,  1006.56,
1002.42,  997.431,  1005.47,  992.773,  1007.92,  1007.36,  999.482,
1003.88,  998.68,  1012.77,  1008.83,  1002.04,  1003.13,  1009.78,
997.305,  1003.33,  1001.45,  1004.12,  999.9,  1002.25,  996.054,
994.309,  1006.29,  999.89,  1003.39,  995.42,  990.872,  1000.26,
994.472,  997.125,  998.633,  994.745,  1001.02,  996.873,  1001.29,
 ... ]
```

10 nilai pertama x adalah

```
>x[1:10]
```

```
[1001.46,  1002.83,  995.424,  1010.42,  993.922,  995.65,  1002.73,
996.528,  998.885,  999.782]
```

Distribusi dapat kita plot dengan flag >distribution dari plot2d.

```
>plot2d(x,>distribution);  ...
> plot2d("qnormal(x,1000,5)",color=red,thickness=2,>add):
```

kita juga dapat mengatur jumlah interval untuk distribusi menjadi 100. Kemudian kita akan melihat seberapa dekat kecocokan distribusi yang diamati dan distribusi yang sebenarnya. Bagaimanapun, kita telah menghasilkan satu juta kejadian.

```
>plot2d(x,distribution=100); ...
>plot2d("qnormal(x,1000,5)",color=red,thickness=2,>add):
```

kita dapat menghitung nilai rata-rata simulasi dan deviasinya harus sangat dekat dengan nilai yang diharapkan.

```
>mean(x), dev(x)
```

```
  999.996936044
  4.99872526095
```

rumus nilai rata rata

$$mean = \frac{\sum x_i}{n}$$

```
>xm=sum(x)/n
```

```
  999.996936044
```

Rumus simpangan percobaannya (deviasi)

$$deviasi = \sqrt{\frac{\sum (x - xm)^2}{n - 1}}$$

```
>sqrt(sum((x-xm)^2/(n-1)))
```

```
  4.99872526095
```

Perhatikan bahwa x-xm adalah vektor nilai yang dikoreksi, di mana xm dikurangi dari semua elemen vektor x.

Berikut adalah 10 nilai pertama x-xm.

```
>short (x-xm)[1:10]
```

```
 [1.4664,  2.8367,  -4.573,  10.419,  -6.075,  -4.3474,  2.7294,
-3.4693,  -1.1123,  -0.21472]
```

Dengan menggunakan bahasa matriks, kita dapat dengan mudah menjawab pertanyaan lainnya. Misalnya, kita ingin menghitung proporsi x yang melebihi 1015.
Ekspresi x>=1015 menghasilkan vektor 1 dan 0. Menjumlahkan vektor ini menghasilkan jumlah kali x[i]>=1015 terjadi.

```
>sum(x>=1015)/n
```

```
 0.001339
```

Probabilitas yang diharapkan dari hal ini dapat dihitung dengan fungsi normaldis(x). sehingga,

$$normaldis(c, m, s) = P(X \leq c)$$

dimana X terdistribusi secara normal m-s.

```
>1-normaldis(1015,1000,5)
```

```
  0.00134989803163
```

cara kerja >distribution flag dari plot2d adalah menggunakan fungsi histo(x), yang menghasilkan histogram frekuensi nilai dalam x. Fungsi ini mengembalikan batas interval dan jumlah dalam interval ini. Kami menormalkan jumlah untuk mendapatkan frekuensi.

```
>{t,s}=histo(x,40); plot2d(t,s/n,>bar):
```

Fungsi histo() juga dapat menghitung frekuensi dalam interval yang diberikan.

```
>{t,s}=histo(x,v=[950,980,990,1010,1020,1050]); t, s,
```

```
  [950,  980,  990,  1010,  1020,  1050]
  [25,  22795,  954608,  22545,  27]
```

hasil tersebut merupakan semua nilai acak yang berada antara 950 dan 1050.

menghitung total jumlah nilai dalam s, yang sama dengan total jumlah elemen dalam x

```
>sum(s)
```

```
  1000000
```

kita akan mensimulasikan 1000 kali lemparan 3 dadu, dan menanyakan pembagian jumlahnya.

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

```
  [3,  18,  23,  45,  65,  106,  108,  137,  136,  133,  88,  65,  36,
  20,  11,  6]
```

kita akan plot hasil tersebut

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

kita akan menggunakan rekursi tingkat lanjut.
Fungsi berikut menghitung banyaknya cara bilangan k dapat direpresentasikan sebagai jumlah dari n bilangan dalam rentang 1 sampai m.

```
>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
```

Berikut hasil pelemparan dadu sebanyak lima kali.

```
>countways(5:25,5,5)
```

```
  [1,  5,  15,  35,  70,  121,  185,  255,  320,  365,  381,  365,  320,
  255,  185,  121,  70,  35,  15,  5,  1]
```

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

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

Kita akan menambahkan nilai yang diharapkan ke plot.

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

Untuk simulasi lain, deviasi 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)')
```

```
  0.317942172219
```

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

Median dari 10 bilangan acak berdistribusi normal 0-1 mempunyai deviasi yang lebih besar.

Karena kita dapat dengan mudah menghasilkan jalan acak, kita dapat mensimulasikan proses Wiener. Kami mengambil 1000 langkah dari 1000 proses. Kami kemudian memplot deviasi standar dan rata-rata langkah ke-n dari proses ini bersama dengan nilai yang diharapkan berwarna 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):
```

uji chi-kuadrat

uji chi-kuadrat adalah alat penting dalam statistik. Di Euler, banyak tes yang diterapkan. Semua pengujian ini mengembalikan kesalahan yang kita terima jika kita menolak hipotesis nol.

Misalnya, kami menguji lemparan dadu untuk distribusi yang seragam. Pada 600 kali lemparan, kami mendapatkan nilai berikut, yang kami masukkan ke dalam uji chi-kuadrat.

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

```
  0.498830517952
```

Ini adalah nilai p-value dari uji chi-kuadrat

Uji chi-kuadrat juga memiliki mode yang menggunakan simulasi Monte Carlo untuk menguji statistiknya,menggunakan Parameter >p menafsirkan vektor y sebagai vektor probabilitas.

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

```
  0.505
```

Ini adalah p-value dari uji chi-kuadrat menggunakan pendekatan Monte Carlo.Dengan simulasi Monte Carlo, kita memperoleh p-value yang mirip dengan uji chi-kuadrat standar (0,4988 di uji pertama)

Selanjutnya kita menghasilkan 1000 lemparan dadu menggunakan generator angka acak, dan melakukan tes yang sama.

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

```
  0.558957010759
```

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

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

```
  0.25577040307
```

Fungsi ttest() memerlukan nilai mean, deviasi, jumlah data, dan nilai mean yang akan diuji.

Sekarang mari kita periksa dua pengukuran untuk mean yang sama. Kami menolak hipotesis bahwa keduanya mempunyai mean yang sama, jika hasilnya <0,05.

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

```
  0.4848094938
```

Jika kita menambahkan bias pada satu distribusi, kita akan mendapatkan lebih banyak penolakan. Ulangi simulasi ini beberapa kali untuk melihat efeknya.

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

```
  0.000205842407976
```

Menambah nilai 2 ke salah satu distribusi menyebabkan p-value menjadi sangat kecil.

Pada contoh berikutnya, kita membuat 20 lemparan dadu acak sebanyak 100 kali dan menghitung yang ada di dalamnya. Rata-rata harus ada 20/6=3,3.

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

```
  3.46
```

Sekarang kita bandingkan jumlah satuan dengan distribusi binomial. Pertama kita plot distribusinya.

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

kita akan Menghitung frekuensi kemunculan setiap jumlah angka ”1” dalam 20 lemparan dadu acak yang telah dilakukan 100 kali

```
>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 merupakan distribusi binomial, jika hasilnya <0,05.

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

```
0.855843823605
```

Contoh berikut berisi hasil dua kelompok orang (misalnya laki-laki dan perempuan) yang 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

Kita akan menguji independensi suara dari jenis kelamin.

```
>tabletest(A)
```

```
  0.990701632326
```

Berikut ini 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
```

uji F

Selanjutnya kita menggunakan analisis varians (uji F) untuk menguji tiga sampel data yang berdistribusi normal untuk nilai mean yang sama. Metode tersebut disebut ANOVA (analisis varians). Di Euler, fungsi varanalisis() 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
```

Dengan p-value sebesar 0.0138 (1,38%), kita bisa menolak hipotesis bahwa ketiga sampel memiliki mean yang sama pada tingkat signifikansi 5% (0.05) dan bahkan pada tingkat signifikansi 1% (0.01). Artinya, terdapat perbedaan yang signifikan antara mean dari setidaknya satu sampel.

Ada juga uji median, yang menolak sampel data dengan distribusi rata-rata yang berbeda, menguji median dari sampel yang disatukan.

```
>a=[56,66,68,49,61,53,45,58,54]
```

```
  [56,  66,  68,  49,  61,  53,  45,  58,  54]
```

```
>b=[72,81,51,73,69,78,59,67,65,71,68,71]
```

```
  [72,  81,  51,  73,  69,  78,  59,  67,  65,  71,  68,  71]
```

```
>mediantest(a,b)
```

```
  0.0241724220052
```

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

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

```
  0.00199969612469
```

Pada contoh berikut, kedua distribusi mempunyai mean yang sama.

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

```
  0.162380065554
```

ini menunjukkan bahwa perbedaan tidak cukup signifikan pada tingkat signifikansi 5%, sehingga hipotesis bahwa kedua distribusi memiliki median yang sama tidak dapat ditolak.

Sekarang 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, apakah a lebih baik dari b.

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

```
  0.0546875
```

Ini kesalahan yang terlalu besar untuk menolak hipotesis. Kita tidak dapat menolak bahwa a sama baiknya dengan b,Karena $p > 0.05$.

Uji Wilcoxon lebih tajam dibandingkan uji ini, namun mengandalkan 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)
```

```
  9.45060349657e-05
```

ini menunjukkan bahwa ada perbedaan signifikan antara kedua sampel pada tingkat signifikansi 5%.

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

```
  0.559353645673
```

hasil ini jauh di atas 0.05, sehingga kita tidak bisa menolak hipotesis bahwa kedua sampel berasal dari distribusi yang sama.

Angka Acak

Berikut ini adalah pengujian pembangkit bilangan acak. Euler menggunakan generator yang sangat bagus, jadi kita tidak perlu mengharapkan adanya masalah.

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

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

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

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

Terakhir, 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;
```

Kami ingin menghitung nilai rata-rata dan simpangan baku yang diukur.

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

```
  999.9
  2.72641400622
```

Kita dapat membuat diagram kotak dan kumis untuk data tersebut. Dalam kasus kita, 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 dalam Euler diakhiri dengan ...dis dan menghitung distribusi probabilitas kumulatif (CPF).

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 tertentu.

```
>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 dapat memasukkan data mentah tersebut ke dalam tabel.

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

Tabel dapat dicetak dengan tajuk. Kita menggunakan vektor string untuk mengatur tajuk.

```
>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 memerlukan nilai rata-rata dan statistik ukuran lainnya, kita perlu menghitung titik tengah rentang. Kita dapat menggunakan dua kolom pertama tabel kita untuk ini.

```
>(T[,1]+T[,2])/2
```

```
157.5
161.5
165.5
169.5
173.5
```

```
177.5
181.5
185.5
```

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

```
>l=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 rata-rata dan deviasi sampel dengan frekuensi yang diberikan.

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

```
  169.901234568
  5.98912964449
```

Mari kita tambahkan distribusi normal nilai-nilai tersebut ke plot.

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

LATIHAN

1.Diberikan data pengukuran tinggi badan pada kelas matematika B adalah sebagai berikut:

Rentang Tinggi (cm)	Jumlah Orang
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

a.)Hitung rata-rata dan deviasi standar dari distribusi tinggi badan ini.
b.)Plot distribusi frekuensi data (diagram batang).
c.)Tambahkan kurva distribusi normal untuk dibandingkan dengan data.

```
>r = 155.5:4:187.5  //Rentang ukuran tinggi badan
```

```
  [155.5,  159.5,  163.5,  167.5,  171.5,  175.5,  179.5,  183.5,  187.5]
```

```
>v = [22, 71, 136, 169, 139, 71, 32, 8] //Jumlah orang dalam tiap rentang
```

```
  [22,  71,  136,  169,  139,  71,  32,  8]
```

```
>l=fold(r,[0.5,0.5])  //Menghitung titik tengah dari setiap rentang tinggi badan
```

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

```
>{m,d}=meandev(l,v); m, d,  //Hitung rata-rata dan deviasi standar
```

```
  169.901234568
  5.98912964449
```

```
>plot2d(r, v, a=150, b=200, c=0, d=190, bar=1, style="\/"):
>plot2d("qnormal(x, m, d) * sum(v) * 4", ...
>xmin=min(r), xmax=max(r), thickness=3, add=1):
>&remvalue();
```

2. Sebuah survei dilakukan untuk mengetahui jumlah jam belajar siswa SMA dalam satu minggu. Berikut data jam belajar dari 10 siswa: 8, 10, 7, 6, 9, 10, 11, 9, 8, 12.
a) Hitung nilai rata-rata dari data di atas
b) Tentukan median dari data tersebut.

```
>M=[8,10,7,6,9,10,11,9,8,12];
>mean(M)
```

```
9
```

```
>median(M)
```

```
9
```

3. Anda diberikan data yang menunjukkan jumlah penjualan barang selama 12 bulan dalam satu tahun berturut-turut 120, 135, 150, 160, 170, 180, 190, 210, 200, 220, 230, 240.
a) Buatlah plot garis dari data penjualan barang tersebut.
b) Hitung rata-rata penjualan perbulan.

```
>X=[120,135,150,160,170,180,190,210,200,220,230,240]
```

```
[120,  135,  150,  160,  170,  180,  190,  210,  200,  220,  230,  240]
```

```
>Y=[1,2,3,4,5,6,7,8,9,10,11,12]
```

```
[1,  2,  3,  4,  5,  6,  7,  8,  9,  10,  11,  12]
```

```
>statplot(Y,X,"l"):
>mean(X)
```

```
183.75
```

Jelasnya, 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. Bagaimanapun, EMT hadir dengan paket numerik dan sistem aljabar komputer.

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

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

Note that this is a work in progress.

Sintaks Dasar

Hal pertama yang Anda pelajari di R adalah membuat vektor. Dalam EMT, perbedaan utamanya adalah operator : dapat mengambil ukuran langkah. Selain itu, ia mempunyai daya ikat yang 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]
```

```
>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 digantikan oleh fungsi seq() di R. Kita dapat 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]
```

```
>function seq(a,b,c) := a:b:c; ...
>seq(0,-0.5,-5)
```

```
  [0,  -0.5,  -1,  -1.5,  -2,  -2.5,  -3,  -3.5,  -4,  -4.5,  -5]
```

```
>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]
```

Fungsi rep() dari R tidak ada di EMT. Untuk masukan vektor dapat dituliskan sebagai berikut.

Perhatikan bahwa "=" atau ":=" digunakan untuk tugas. Operator "->" digunakan untuk satuan dalam EMT.

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

```
  77.6713990297 miles
```

Operator "<-" untuk penugasan memang bukan ide yang baik untuk R.

tetapi di EMT operator "<-" itu bukan penugasan melainkan perbandingan
Berikut ini akan membandingkan a dan -4 di EMT.

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

```
  0
```

EMT dan R memiliki vektor bertipe 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 memunculkan kesalahan atau menghasilkan NAN tergantung pada tanda "kesalahan".

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

```
  NAN
  1
```

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

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

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

```
© René Grothmann
```

karakter khusus (hak cipta © dan karakter aksen é),

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

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

Penggabungan string dilakukan dengan "+" atau "|". Penggabungan ini akan menghasilkan string tunggal, dan angka yang digabungkan akan dikonversi otomatis ke format string. Ini dapat mencakup angka, yang akan dicetak dalam format saat ini.

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

```
pi = 3.14159265359
```

Pengindeksan

Seringkali, ini akan berfungsi seperti di R.

Namun EMT akan menafsirkan indeks negatif dari belakang vektor, sementara R menafsirkan 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
```

```
>x, x[1:5], x[-3]
```

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

Untuk meniru perilaku R di EMT, kita dapat menggunakan fungsi drop(x,n)

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

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

Vektor logika tidak diperlakukan berbeda sebagai indeks di EMT, berbeda dengan R. Anda perlu mengekstrak 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]
```

Tipe Data

EMT memiliki lebih banyak tipe data tetap daripada R. Jelasnya, di R terdapat vektor yang berkembang. Anda dapat mengatur vektor numerik kosong v dan memberikan nilai ke elemen v[17]. Hal ini tidak mungkin dilakukan di EMT.

Berikut ini agak tidak efisien.

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

kenapa cara ini kurang efisien? karna setiap elemen baru di tambahkan EMT harus menyalin selurus isi v kembali ke variabel v.

Semakin efisien vektor telah ditentukan sebelumnya.

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

Untuk mengubah tipe data di EMT, Anda dapat menggunakan fungsi seperti kompleks().

```
>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
```

- Variabel s diinisialisasi sebagai string "[" untuk menyimpan hasil akhir. Awalnya, tanda kurung buka [ditambahkan ke variabel s sebagai pembuka.
- loop 1 to length(v); menjalankan perulangan dari elemen pertama hingga elemen terakhir dalam v. Fungsi length(v) mengembalikan panjang atau jumlah elemen dalam vektor v.
- print(v[], 2, 0); adalah fungsi format yang mengonversi elemen vektor v pada posisi saat ini (v[]) menjadi string.
parameter 2 menunjukkan bahwa dua digit setelah titik desimal akan ditampilkan, sementara 0 memastikan bahwa angka ditampilkan tanpa tambahan simbol atau format lainnya.
- Bagian if <length(v) memeriksa apakah elemen saat ini bukan elemen terakhir. Jika benar, maka koma , akan ditambahkan ke variabel s untuk memisahkan elemen.
- Setelah loop selesai, tanda kurung tutup] ditambahkan ke string s, dan string ini kemudian dikembalikan sebagai output.

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

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

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

Untuk Latex perintah tex dapat digunakan untuk mendapatkan perintah Latex.

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

$$[1, 2, 3]$$

Faktor dan Tabel

Dalam pengantar R ada contoh yang disebut faktor.

Berikut ini adalah daftar wilayah 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"];
```

Perintah diatas digunakan untuk mendefinisikan sebuah array (array sendiri adalah sekumpulan variabel yang memiliki tipe data yang sama) karena pada data tersebut ada beberapa nama negara bagian yang terulang. Array ini berisi singkatan untuk negara bagian dan teritori di Australia.

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 mari kita coba mencari nilai mean dan median dari data pendapatan tersebut menggunakan perintah mean(incomes) dan median(incomes)

```
>mean(incomes)
```

```
  54.7333333333
```

```
>median(incomes)
```

```
  57
```

Sekarang, kami ingin menghitung rata-rata pendapatan di suatu wilayah. Menjadi program statistik, R memiliki faktor() dan tappy() untuk ini.

EMT dapat melakukan hal ini dengan menemukan indeks wilayah dalam daftar wilayah unik.

```
>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 titik itu, kita dapat menulis fungsi perulangan 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
```

i: Parameter pertama biasanya adalah nilai yang digunakan untuk pencocokan atau pemetaan.
f$:call: Parameter kedua, yang kemungkinan besar adalah sebuah fungsi yang dipanggil dalam kode tersebut. f$ di sini merujuk pada fungsi yang diterima sebagai input.
cat: Parameter ketiga adalah array atau vektor yang berisi kategori yang akan diproses.
x: Parameter keempat adalah array atau vektor yang akan diproses atau diubah berdasarkan pemetaan kategori yang dilakukan.

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

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

```
[44.5,  57.3333333333,  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,  57.3333333333]
```

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

Sehingga kita bisa mencetak rata-rata pendapatan negara di daerah secara 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
```

Fungsi writetable digunakan untuk menampilkan data dalam bentuk tabel yang terstruktur dengan label kolom dan lebar kolom yang dapat disesuaikan.
Dengan labc=auterr, berarti menetapkan label kolom untuk tabel tersebut berdasarkan kategori yang ada di auterr(yang sudah diurutkan sesuai abjad).
wc(width of columns)=7 berarti setiap kolom dalam tabel akan memiliki lebar minimal 7 karakter.
sebagai contoh 44.5 itu memiliki 4 karakter (termasuk titik desimal).
karena data dalam kolom lebih pendek dari 7 karakter, kolom tersebut diberi ruang ekstra untuk tampilan yang rapi.

Kita juga bisa mencoba meniru perilaku R sepenuhnya.

Faktor-faktor tersebut harus disimpan dengan jelas dalam kumpulan beserta jenis dan kategorinya (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);
```

Perintah statef = makef(austates); digunakan untuk mengolah data yang ada dalam variabel austates, dan mengidentifikasi elemen unik yang ada dalam data tersebut.

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]
```

statef[3] adalah elemen ketiga dari koleksi yang dikembalikan oleh fungsi makef, yaitu indeks posisi dari elemen-elemen dalam austates yang sudah dipetakan ke urutan dalam u (data unik yang terurut). statef[3] akan mengembalikan indeks posisi dari setiap elemen dalam austates berdasarkan urutan yang ada di u.

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) yang tidak memiliki data. Tetapi kita harus memeriksa panjang t yang benar dan kebenaran pengumpulan 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 perpustakaan 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
```

Fungsi shortformat digunakan untuk mengatur format tampilan angka agar lebih ringkas dan mudah dibaca.
Perintah diatas digunakan untuk membuat matrik X dari angka 1 sampai 20 dengan ketentuan matriks dengan 4 baris dan 5 kolom.

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

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

```
 2  3
 7  8
12 13
17 18
```

Perintah diatas digunakan untuk menampilkan matriks X kolom kedua sampai ketiga.

```
>X[,3:5]
```

```
 3  4  5
 8  9 10
13 14 15
18 19 20
```

Namun, di R dimungkinkan untuk menyetel daftar indeks vektor tertentu ke suatu nilai. Hal yang sama mungkin terjadi 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
```

Perintah setmatrixvalue(M, i, j, v) adalah fungsi yang digunakan untuk mengubah nilai elemen-elemen dalam matriks berdasarkan indeks tertentu.
M: Matriks yang akan dimodifikasi.

i: Indeks baris atau posisi baris dalam matriks M yang ingin diubah.

j: Indeks kolom atau posisi kolom dalam matriks M yang ingin diubah.

v: Nilai yang akan dimasukkan ke dalam elemen-elemen matriks M pada posisi yang ditentukan oleh indeks i dan j.

Kami mendemonstrasikan ini untuk menunjukkan bahwa matriks dilewatkan dengan referensi di EMT. Jika Anda tidak ingin mengubah matriks M asli, 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
```

Perkalian luar dalam EMT hanya dapat dilakukan antar vektor. Ini otomatis karena bahasa matriks. Satu vektor harus berupa vektor kolom dan vektor lainnya harus berupa 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
```

1:5: Ini adalah vektor baris yang berisi angka-angka dari 1 hingga 5
(1:5)': Tanda ' di sini menunjukkan transposisi dari vektor baris 1:5. Dengan kata lain, ini mengubah vektor baris menjadi vektor kolom.

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

Tentu saja, hal ini dapat 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.

Namun kita ingin meniru perilaku R. Untuk melakukannya, kita 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 multiplisitas eksak, EMT dapat menghitung frekuensi dalam vektor.

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

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

Perintah diatas digunakan untuk menghitung distribusi frekuensi nilai-nilai dalam vektor q dalam rentang dari -50 hingga 50, dengan interval 10. Fungsi ini menghitung berapa banyak nilai dalam q yang jatuh dalam setiap interval: [-50, -40), [-40, -30), ..., [40, 50).

Cara paling mudah untuk memplotnya sebagai distribusi adalah sebagai berikut.

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

Namun dimungkinkan juga untuk menghitung terlebih dahulu penghitungan 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="/"):
```

Daftar

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.

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

1. Membuat list dan mengakses elemen dalam list

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

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

Perintah diatas digunakan untuk membuat list L dengan nama depan Fred, nama belakang Flintstone, usia 40, dan tahun 1990, 1992.
Namun untuk tahun tersebut tidak dapat dipastikan apa arti dari tahun-tahun tersebut, bisa saja tahun kelahiran dan kematian, tahun pendidikan, tahun pekerjaan, atau yang lainnya.

Saat ini unsur-unsur tersebut tidak memiliki nama, meskipun nama dapat ditetapkan untuk tujuan khusus. Mereka diakses dengan nomor.

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

```
1992
```

Perintah diatas digunakan untuk menampilkan list L keempat urutan kedua. Karena pada list L keempat berisi tahun yang dimana terdapat 2 tahun, tahun pertama adalah 1990 dan tahun kedua adalah 1992. Perintah tersebut ingin menampilkan tahun kedua, maka outputnya adalah 1992.

2. Menggabungkan dua list

```
>A := [1,2,3]
```

```
  [1,  2,  3]
```

```
>B := [4,5,6]
```

```
  [4,  5,  6]
```

```
>C := [A, B]
```

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

3. Mengubah elemen dalam list

```
>D := [7,8,9,10]
```

```
[7,  8,  9,  10]
```

```
>D[3] := 99
```

```
[7,  8,  99,  10]
```

4. menghitung panjang list

```
>E := [10,20,30,40,50,60,70]
```

```
[10,  20,  30,  40,  50,  60,  70]
```

```
>len := length(E)
```

```
7
```

File Input dan Output (Membaca dan Menulis Data)

Anda sering kali ingin mengimpor matriks data dari sumber lain ke EMT. Tutorial ini memberi tahu Anda tentang banyak cara untuk mencapai hal ini. Fungsi sederhananya adalah writematrix() dan readmatrix().

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

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

```
  0.44779
  0.29446
```

$$mean = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$dev = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - x)^2}$$

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

Karena pengenalan 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, kita menggunakan readmatrix()

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

```
[0.74157,  0.13184,  0.082656,  0.1975,  0.63019,  0.51127,  0.98091,
0.020501,  0.29063,  0.83916,  0.051539,  0.45459,  0.77198,  0.8754,
0.49257,  0.96765,  0.48685,  0.11645,  0.086971,  0.35536,  0.69515,
0.19161,  0.66654,  0.61295,  0.38779,  0.16748,  0.65044,  0.64784,
0.17712,  0.048322,  0.24795,  0.8682,  0.49634,  0.9125,  0.079817,
0.6544,  0.66279,  0.54949,  0.10807,  0.71727,  0.75473,  0.34844,
0.38634,  0.40978,  0.3234,  0.32117,  0.85977,  0.78211,  0.12933,
0.39591,  0.018822,  0.95697,  0.26985,  0.2409,  0.52222,  0.79655,
0.30484,  0.53489,  0.014971,  0.22246,  0.13021,  0.82292,  0.83054,
0.99006,  0.064585,  0.57221,  0.16539,  0.31818,  0.16037,  0.71343,
0.6188,  0.67365,  0.22367,  0.86965,  0.67439,  0.859,  0.85639,
0.07405,  0.65051,  0.79298,  0.039088,  0.4536,  0.24248,  0.10425,
0.36493,  0.084869,  0.37012,  0.045487,  0.66347,  0.69294,  0.46928,
0.27625,  0.20081,  0.069966,  0.2009,  0.52929,  0.97226,  0.52319,
0.11857,  0.075214]
```

Dan hapus file tersebut.

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

```
  0.44779
  0.29446
```

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

Misalnya, jika Anda memiliki sistem Indonesia (titik desimal dengan koma), Excel Anda memerlukan nilai dengan koma desimal yang dipisahkan dengan titik koma dalam file csv (defaultnya adalah nilai yang dipisahkan koma). File berikut "test.csv" akan muncul di folder saat ini 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 kita memiliki string dengan token seperti berikut.

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

Untuk melakukan tokenisasi 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
0.5894214399628446,0.1500948494104958,0.4911562128682984
0.2817236329179764,0.3586251183800097,0.1159031925964036
0.2135992153342503,0.3457575959104093,0.3271359940090028
```

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

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

Koleksi ini dapat dicetak dengan writetable() ke buku catatan, atau ke file.

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

```
         A         B         C
   0.58942   0.15009   0.49116
   0.28172   0.35863    0.1159
    0.2136   0.34576   0.32714
```

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

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

```
   0.41022
   0.25208
    0.2955
```

File CSV

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

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

Here is the content of this file.

```
>printfile(file)
```

```
0.5537619367566726,0.6738419762237779,0.04291882768650555
0.2401989793483695,0.0861139922960146,0.3678722550483149
0.1343166294586442,0.3616257867488041,0.3383621511798272
```

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

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

```
>readmatrix(file)
```

```
    0.55376   0.67384  0.042919
     0.2402  0.086114   0.36787
    0.13432   0.36163   0.33836
```

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 "save as" dan "other format", lalu pilih "CSV". Pastikan tabel saat ini hanya berisi data yang ingin Anda ekspor.

Ini sebuah contoh.

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

```
Could not open the file
excel-data.csv
for reading!
Try "trace errors" to inspect local variables after errors.
printfile:
    open(filename,"r");
```

Seperti yang Anda lihat, sistem bahasa Jerman saya menggunakan titik koma sebagai pemisah dan koma desimal. Anda dapat mengubahnya di pengaturan sistem atau di Excel, tetapi hal ini tidak diperlukan untuk membaca matriks menjadi EMT.

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

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

```
Could not open the file
excel-data.csv
for reading!
Try "trace errors" to inspect local variables after errors.
readmatrix:
    if filename<>"" then open(filename,"r"); endif;
```

Let us plot this.

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

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

Fungsi berikut adalah contohnya. 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)
```

```
 0.55376         0   0.67384         0  0.042919
  0.2402         0  0.086114         0   0.36787
 0.13432         0   0.36163         0   0.33836
```

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

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

```
 0.55376         0   0.67384
       0  0.042919    0.2402
       0  0.086114         0
```

Oleh karena itu sangat mudah untuk menyimpan suatu vektor nilai, satu nilai di setiap baris dan membaca kembali vektor ini.

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

```
  0.493
```

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

```
  0.493
```

Tabel dapat digunakan untuk membaca atau menulis data numerik. Misalnya, kita menulis tabel dengan header baris dan kolom ke sebuah file.

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

```
one,two,three
      0.56,      0.63,      0.59
      0.45,      0.86,      0.44
      0.72,      0.19,      0.91
```

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
       0.56      0.63      0.59
       0.45      0.86      0.44
       0.72      0.19      0.91
```

Menganalisis Garis

Pada subbab ini sering digunakan untuk memproses atau mengekstrak data dari teks yang berformat khusus, seperti data tabel dallam HTML. Anda bahkan dapat mengevaluasi setiap baris dengan tangan. Misalkan, kita memiliki baris dengan format berikut.

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

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

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

Pertama, kita akan memisahkan string line menjadi bagian-bagian yang lebih kecil, yang dikenal sebagai "token".

```
>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],"'","")();
```

Dengan menggunakan ekspresi reguler, dimungkinkan untuk mengekstrak hampir semua informasi dari sebaris data.

Selanjutnya, kita akan melihat bagaimana mengekstrak data string yang berisi markup HTML menggunakan ekspresi reguler.

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

Untuk mengekstraknya, kami menggunakan ekspresi reguler, yang mencari

- tanda kurung tutup >, untuk mengindikasikan bahwa kita akan

mencari awal dari elemen yang ada di dalam tag.

- string apa pun yang tidak mengandung tanda kurung akan mencocokkan

elemen di dalam tag <td>.

- braket pembuka dan penutup menggunakan solusi terpendek,dengan tag

pembuka (<td>) dan penutup (</td>).

- sekali lagi string apa pun yang tidak mengandung tanda kurung,ini

akan menjamin bahwa kita akan mengambil isi yang relevan di dalam tagnya.

- dan tanda kurung buka < menandai bahwa ini adalah akhir dari tag

dan awal dari tag baru.
Mencari pola tertentu dalam string line yang menggunakan ekspresi reguler.

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

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

Kita akan mengeksekusi elemen-elemen di dalam array atau list vt satu per satu dalam sebuah perulangan.

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

```
1145.5
5.6
```

Berikut 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
```

Kita akan mengekstrak dan menampilkan semua nilai yang berada di antara tag <td>...</td> dalam baris,dan mencari apakah nilai tersebut numerik atau bukan.

```
>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, kita membaca versi terkini dari situs EMT. Kami menggunakan ekspresi reguler untuk memindai "Versi ..." dalam sebuah 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
```

Contoh lain membaca URL dengan EMT
”https://mywebsite.com/version.h”

```
>function readversionmywebsite () ...
```

```
urlopen("https://mywebsite.com/version.h");
repeat
   until urleof();
   s=urlgetline();
   k=strfind(s,"Release",1);
   if k>0 then substring(s,k,strfind(s,"<",k)-1); break; endif;
end;
urlclose();
endfunction
```

```
>readversionmywebsite
```

Karena string ”Release” tidak ada di dalam file version.h, maka strfind(s, ”Release”, 1) akan mengembalikan nilai nol atau tidak menghasilkan indeks yang diperlukan untuk proses pencarian.

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

```
  M = [ ..
  0.3888176479607355, 0.3870125417113612;
  0.2576115973745406, 0.1750949536837037];
```

Sekarang kita dapat memuat file tersebut. Ini akan mendefinisikan matriks M.

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

```
M =
  0.38882   0.38701
  0.25761   0.17509
```

Omong-omong, jika writevar() digunakan pada suatu variabel, definisi variabel dengan nama variabel tersebut akan dicetak.

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

```
M = [ ..
0.3888176479607355, 0.3870125417113612;
0.2576115973745406, 0.1750949536837037];
inch$ = 0.0254;
```

Kita juga bisa membuka file baru atau menambahkan file yang sudah ada. Dalam contoh kita menambahkan file yang dibuat sebelumnya.

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

```
M1 =
  0.17881  0.026345
  0.63559   0.34332
M2 =
  0.22274
  0.91626
   0.4616
```

Untuk menghapus file apa pun, gunakan fileremove().

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

Vektor baris dalam suatu file tidak memerlukan koma, jika setiap angka berada pada baris baru. Mari kita buat file seperti itu, tulis 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 = [
0.743886392671
0.92708388148
0.46506977432
0.0265574069961
0.147047179518
];
```

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

```
[0.74389,  0.92708,  0.46507,  0.026557,  0.14705]
```

LATIHAN

1. Misalkan anda memiliki vektor x=[2,4,6,8,10]
a. buatkan vektor yang menggabungkan vektor x,angka0dan vektorx lagi
b. tentukan apakah setiap elemen vektor x lebih besar dari 5(hasil logika 1 untuk benar dan 0 untuk salah)

```
>x:=[2,4,6,8,10]; [x,0,x]
```

```
  [2,  4,  6,  8,  10,  0,  2,  4,  6,  8,  10]
```

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

```
  [0,  0,  1,  1,  1]
  [0,  0,  6,  8,  10]
```

2. Tentukan matriks X dengan elemen-elemen yang berurutan dari 1 hingga 20 dan susunlah elemen tersebut menjadi matriks berukuran 5x4.

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

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

3.Seorang analis memiliki data penjualan harian selama 5 hari(150,200,250,300,350) yang disimpan dalam bentuk vektor sebagai berikut:
a. mean(rata-rata)
b. deviasi standar

```
>penjualan=[150,200,250,300,350]
```

```
  [150,  200,  250,  300,  350]
```

atau anda bisa memanggil data yang sudah dibuat

```
>filename="penjualan.dat";
>writematrix(penjualan',filename)
>penjualan=readmatrix(filename)'
```

```
  [150,  200,  250,  300,  350]
```

```
>mean(penjualan)
```

```
  250
```

```
>dev(penjualan)
```

```
79.057
```

4. Buat fungsi yang membuka URL
"https://en.wikipedia.org/wiki/Euler_(software)"
dan mencari kata "Versi" di dalam URL tersebut, dan tampilkan hasilnya.

```
>function readversionwebsite () ...
```

```
urlopen("https://en.wikipedia.org/wiki/Euler_(software)");
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
```