7. Loops and Conditional Branching

2015-04-27

```
$Version
```

```
10.0 for Mac OS X x86 (64-bit) (September 10, 2014)
```

7.1 Sums and Summation of Infinite Series

`Sum[f, {i,imax}]`	$\sum_{i=1}^{i=\text{imax}}$	f
`Sum[f, {i,imin,imax} ]`	$\sum_{i=\text{imin}}^{i=\text{imax}}$	f
`Sum[f, {i,imin,imax,di}]`	same as above but with increment	di
`Sum[f,{i,imin,imax},{j,jmin,jmax}]`	$\sum_{i=\text{imin}}^{i=\text{imax}} \sum_{j=\text{jmin}}^{j=\text{jmax}}$	f multiple sum
`NSum[...]`	Numeric Sum	

`NSum[...]` need not be the same as **`N[Sum[...]]`**, as in the first command *Mathematica* uses numeric procedures right from the beginning, whereas in the second command *Mathematica* may start with analytic calculations.

```
Sum[k^2, {k,4}]
```

```
30
```

```
f = Sum[1/(i+1), {i,0,4}]
```

$$\frac{137}{60}$$

```
Sum[k^2, {k,n}]
```

$$\frac{1}{6} n\,(1+n)\,(1+2\,n)$$

```
f = Sum[q^n, {n,0,Infinity}]
```

$$\frac{1}{1-q}$$

```
f = Sum[.9^n, {n,0,Infinity}]
```

```
10.
```

```
f = Sum[1/n, {n,0,Infinity}]
```

Power::infy : Infinite expression $\frac{1}{0}$ encountered. >>

$$\sum_{n=0}^{\infty} \frac{1}{n}$$

Power::infy ; Infiniteexpression ∞ encountered.

```
Sum[(-1)^n / n, {n, Infinity}]
```

```
-Log[2]
```

```
Sum[ (n-2k)^2 Binomial[n, k], {k,0,n} ]
```

$2^n n$

```
Sum[1/k^4, {k,Infinity}]
```

$\frac{\pi^4}{90}$

```
Sum[x^k /k! , {k,Infinity} ]
```

$-1 + e^x$

```
Sum[1/k^3, {k,Infinity}]
```

```
Zeta[3]
```

```
?Zeta
```

Zeta[s] gives the Riemann zeta function $\zeta(s)$.
Zeta[s, a] gives the generalized Riemann zeta function $\zeta(s, a)$. ≫

- $\zeta(s) = \sum_{k=1}^{\infty} k^{-s}$.
- $\zeta(s, a) = \sum_{k=0}^{\infty} (k + a)^{-s}$, where any term with $k + a = 0$ is excluded.

```
Zeta[3] // N
```

```
1.20206
```

```
NSum[1 / k^3, {k, Infinity}]
```

```
1.20206
```

```
Sum[x^k / (k ! k), {k, Infinity} ]
```

```
-EulerGamma - Gamma[0, -x] - Log[-x]
```

EulerGamma is Euler' s constant gamma, with numerical value $\gamma =$ 0.577216 ....

```
% /. x -> 1.37 //N//Chop
```

```
2.02865
```

```
NSum[1.37^k /(k! k), {k,Infinity} ]
```

```
2.02865
```

```
Clear[s1]
s1[n_, x_] = Sum[ Sin[(2 i - 1) x] / i, {i, n} ] // Expand
```

$$-\frac{1}{2} i e^{-i x} \left(e^{-2 i x}\right)^n \text{LerchPhi}\left[e^{-2 i x}, 1, 1+n\right] + \frac{1}{2} i e^{-i x} \left(e^{2 i x}\right)^{1+n} \text{LerchPhi}\left[e^{2 i x}, 1, 1+n\right] + \frac{1}{2} i e^{-i x} \text{Log}\left[1 - e^{2 i x}\right] - \frac{1}{2} i e^{i x} \text{Log}\left[e^{-2 i x} \left(-1 + e^{2 i x}\right)\right]$$

```
? LerchPhi
```

LerchPhi[z, s, a] gives the Lerch transcendent $\Phi(z, s, a)$. ≫

It is a mathematical function, suitable for both symbolic and numerical manipulation.

$\Phi(z, s, a) = \sum_{k=0}^{\infty} z^k / (k + a)^s$.

```
s1[5, x] // Simplify // Expand // ExpToTrig
```

$$\text{Sin}[x] + \frac{1}{2}\text{Sin}[3x] + \frac{1}{3}\text{Sin}[5x] + \frac{1}{4}\text{Sin}[7x] + \frac{1}{5}\text{Sin}[9x]$$

```
s1[n, x] /. { x -> 27.31, n -> 5} // Chop
```

```
0.861638
```

```
s1[n, x] /. { x -> 27.31, n -> 5.5} // Chop
```

```
0.796852
```

```
s2[n_,x_] = Sum[ Sin[i x]/i, {i,1,n,2} ]//Expand
```

$$\frac{1}{2} i \,\text{ArcTanh}\left[e^{-i x}\right] - \frac{1}{2} i \,\text{ArcTanh}\left[e^{i x}\right] - \frac{1}{4} i\, e^{-3 i x} \left(e^{-2 i x}\right)^{\text{Floor}\left[\frac{1}{2}(-1+n)\right]} \text{LerchPhi}\left[e^{-2 i x}, 1, \frac{3}{2} + \text{Floor}\left[\frac{1}{2}(-1+n)\right]\right] + \frac{1}{4} i\, e^{3 i x} \left(e^{2 i x}\right)^{\text{Floor}\left[\frac{1}{2}(-1+n)\right]} \text{LerchPhi}\left[e^{2 i x}, 1, \frac{3}{2} + \text{Floor}\left[\frac{1}{2}(-1+n)\right]\right]$$

```
s2[7, .33] // Chop
```

```
0.907664
```

```
s2[7,x]//Simplify//Expand//ExpToTrig
```

$$\text{Sin}[x] + \frac{1}{3}\text{Sin}[3x] + \frac{1}{5}\text{Sin}[5x] + \frac{1}{7}\text{Sin}[7x]$$

```
(% /. x -> .33) // Chop
```

```
0.907664
```

```
Sum[ Sin[k x]/k, {k,Infinity} ]
```

$$\frac{1}{2} i \left(\text{Log}\left[1 - e^{i x}\right] - \text{Log}\left[e^{-i x}\left(-1 + e^{i x}\right)\right]\right)$$

```
Sum[ Sinh[k x] z^k / k, {k, Infinity} ] // Simplify // Apart
```

$$\frac{1}{2}\text{Log}[1 - e^{-x} z] - \frac{1}{2}\text{Log}[1 - e^{x} z]$$

```
f = NSum[(-1)^k/k,  {k,1,Infinity} ]
```

```
-0.693147
```

```
- Log[2.]
```

```
-0.693147
```

```
f = NSum[(-1)^k/(2k+1),  {k,0,Infinity} ]
```

```
0.785398
```

```
Pi/4 //N
```

```
0.785398
```

```
NSum[ 1/Log[k], {k,2,Infinity}]
```

NIntegrate::slwcon :
Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand, or WorkingPrecision too small. ≫

NIntegrate::ncvb : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in k near {k} = $\{8.16907 \times 10^{224}\}$. NIntegrate obtained 1.333846174653637`15.954589770191005*^27945 and 1.333846174653637`15.954589770191005*^27945 for the integral and error estimates. ≫

$1.333846174653637 \times 10^{27\,945}$

In reality the above sum diverges.

```
NSum[(-1)^k /Log[k], {k,2,Infinity}]
```

```
0.9243
```

```
fc[n_, x_] = - (-1)^n Cos[n x] / n
```

$$\frac{(-1)^{1+n}\,\text{Cos}[n\,x]}{n}$$

```
sn[x_] := NSum[fc[n, x // N], {n, Infinity}]
```

```
fe[x_] = Log[2 Cos[x / 2]];
```

```
fe[0] // N
```

```
0.693147
```

```
sn[0]
```

```
0.693147
```

```
fe[Pi / 2] // N
```

```
0.346574
```

```
sn[Pi / 2 // N]
```

$0.346579 - 0.0000151147\,i$

The same troubles also turn up for other non-zero values of x as, e.g. for x = 0.15 .

```
fe[.15]
```

```
0.690332
```

```
sn[.15]
```

$0.690332 + 2.19792 \times 10^{-7}\,i$

```
Sum[(-1)^n, {n, Infinity}]
```

Sum::div : Sum does not converge. ≫

$$\sum_{n}^{\infty} (-1)^n$$

```
NSum[(-1)^n, {n, Infinity}]
```

```
-0.5
```

```
NSum[(-1)^n, {n, 0, Infinity}]
```

```
0.5
```

Common convergence of series

In general, a series comprises an infinity of terms:

r1 + r2 + r3 + ... = $\sum_{n=1}^{N} r_n$ (1)

partial sums = s(N) = $\sum_{n=1}^{N} r_n$

The convergence of this sequence of numbers or functions:

s1 , s2 , s3 , s4 , ..., sn , ...

If this sequence tends toward a limit s, then this is called the sum of the series):

s1 , s2 , s3 , s4 , ..., sn , ... → s (2)

Convergence of Hölder means (arithmetic means of partial sums}

$h_n^{(1)}$ = (s1 + s2 + ... + sn)/N (3)

$lim_{N\to\infty}\ h_N^{(1)}$ = S

If S is finite, the sequence (2) is called limitable.

```
r[n] = (-1)^n ;
s[n] = Sum[r[n], {n, N}]
```

$$\frac{1}{2}\left(-1+(-1)^{N}\right)$$

```
hs[N] = Sum[s[n] / N, {n, N}]
```

$$\frac{1}{2}\left(-1+(-1)^{N}\right)$$

```
hos = Sum[hs[N], {N, NN}] / NN // ExpandAll
```

Simplify::timc : Number of seconds 300 is not a positive machine-sized number or Infinity. »

Simplify::timc : Number of seconds 300 is not a positive machine-sized number or Infinity. »

Simplify::timc : Number of seconds 300 is not a positive machine-sized number or Infinity. »

General::stop : Further output of Simplify::timc will be suppressed during this calculation. »

$$-\frac{1}{2}-\frac{1}{4\,NN}+\frac{(-1)^{NN}}{4\,NN}$$

```
Limit[hos, NN → Infinity]
```

$$-\frac{1}{2}$$

```
f = Sum[q^n, {n,Infinity}]
```

$$-\frac{q}{-1+q}$$

```
f /. q → - 1
```

$$-\frac{1}{2}$$

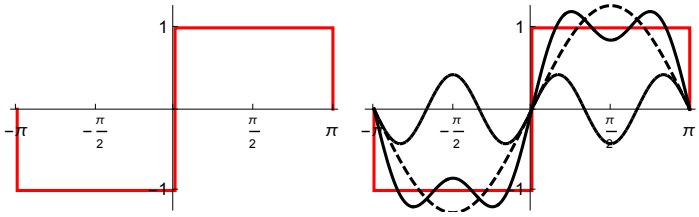
7.1.1 Fourier Series and the Gibb's Phenomenon

In 1899 the Canadian theoretical phycisist W. Gibbs found a peculiar behaviour of Fourier series representing discontinuous functions. To show this peculiarity, called Gibb's phenomenon, the Fourier series belonging to the periodic step function displayed below in the left picture is summed.

```
me = {{-π, 0}, {-π, -1}, {0, -1}, {0, 1}, {π, 1}, {π, 0}};
p1 = ListPlot[me, Joined → True, Ticks → {π Range[-2, 2] / 2, Range[-1, 1]},
   PlotStyle → Hue[0], PlotRange → 1.25 {-1, 1}];
```

```
Clear[n,x]; pb = Black;
yf[n_,x_] = (4/Pi) Sin[n x]/n;
sf[n_,x_] :=  Sum[ yf[i,x], {i,1,n,2}];
ssf  = Plot[ {sf[2,x], yf[3,x], sf[3,x]}, {x,Pi,-Pi},
PlotStyle -> {{pb,Dashing[{.02}]},{pb,Dashing[{.01}]},{pb,Dashing[{}]}},
Ticks->{π Range[-2,2]/2, Range[-1,1]}];

p2 = Show[p1, ssf]; Show[GraphicsRow[{p1, p2}]]
```

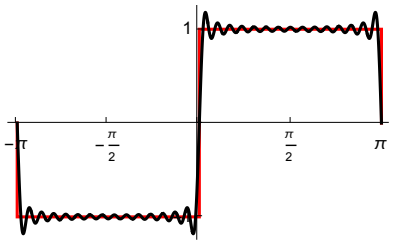


The picture at the right shows the first harmonic (curve with long dashes), the second harmonic (short dashes) and the sum of these two (continuous curve). The first harmonic overshootes the value 1 at $x = \pm\pi/2$. The second harmonic (over)compensates this effect. and leads to overshooting near $x = \pm\pi/4$ and $x = \pm 3\pi/4$. The next harmonic will compensate all these overshootings but will introduce new overshootings at points stlll nearer to the original discontinuities of the step functions. At points in the interior of the intefvalls $(-\pi,0)$ and $(0,\pi)$ this wiggles will smooth out in the limit $N \to \infty$.

$$s(x) \quad = \quad \lim_{N\to\infty} \sum_{n=1}^{N} s_n(x) \quad = \quad \lim_{N\to\infty} \sum_{n=1}^{N} b_n \sin(n\,x)$$

But there remain steep narrow peaks approaching the points where the steps occur, as can be seen in the picture below.

```
p3 = Plot[sf[30, x], {x, -π, π},
    Ticks → {π Range[-2, 2] / 2, Range[-1, 1]}, PlotStyle → pb,
    PlotRange → 1.2 {-1, 1}];
Show[p1, p3, ImageSize → 200]
```



Analytic analysis shows that these overshoots at the jumps remain in the limit $N \to \infty$, but the value of the limit s(x) depends very sensitively upon the way the limits $N \to \infty$ and $x \to 0+$, for example, are taken.
The most important point is: Common summation of Fourier series representing discontinuous functions leads to Gibb's phenomenon. This can be avoided by summing the Fourier series in a different way, namely by Fejer's arithmetic means (corresponding to the Hölder means treated in the previous section) :

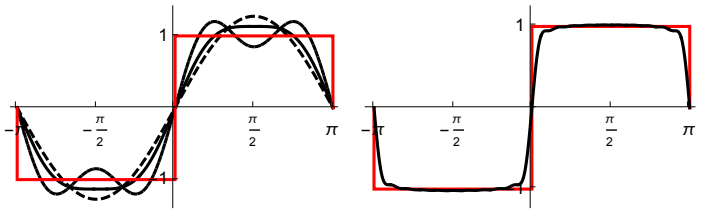
$$s(x) \quad = \quad \lim_{N\to\infty} \sum_{n=1}^{N} \sigma_n(x),$$

$$\sigma_n(x) \quad = \quad n^{-1} \sum_{n=1}^{N} s_i(x) \quad = \quad n^{-1} \sum_{n=1}^{N} \sum_{k=1}^{i} b_k \sin(k\,x)$$

```
sff[n_,x_] :=  Sum[ sf[i,x], {i,n}]/n ;
Plot[Evaluate[{sff[2,x], sf[3,x], sff[3,x]}], {x,Pi,-Pi},
 PlotStyle -> {{pb,Dashing[{.02}],pb},{pb,Dashing[{.01}]},{pb,Dashing[{}]}},
 Ticks → {π Range[-2, 2]/2, Range[-1, 1]}];

ss3 = Show[%,p1];
```

```
p4 = Plot[Evaluate[sff[20, x]], {x, π, -π}, PlotStyle → pb];
ss0 = Show[p1, p4, Ticks → {π Range[-2, 2] / 2, Range[-1, 1]}];
GraphicsRow[{ss3, ss0}]
```



7.1.2 Sums of Fourier Series

Sum[] permits one to get closed expressions for some of the Fourier series as given below. These expressions display the periodic behaviour of the series. Summation of the series by residues (complex analysis) may lead to still simpler expressions (e.g. polynomials), which are valid in a limited interval. Below it is shown, how one may guess these expressions from plots and how these may be generalized to periodic expressions.

```
s1 = Sum[Sin[n x]/n, {n,1,Infinity}]
```

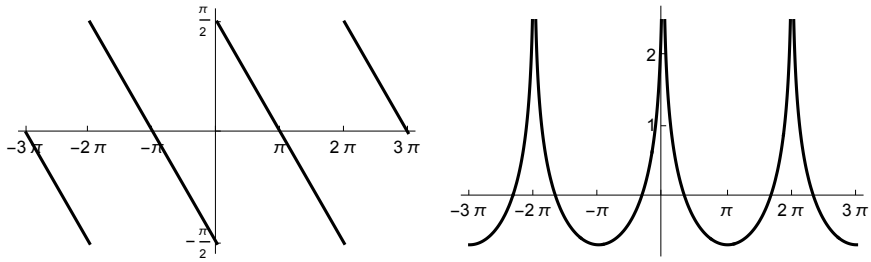
$$\frac{1}{2}\,i\,\left(\text{Log}\left[1-e^{i\,x}\right]-\text{Log}\left[e^{-i\,x}\left(-1+e^{i\,x}\right)\right]\right)$$

```
c1 = Sum[Cos[n x]/n, {n,1,Infinity}]//ExpandAll
```

$$-\frac{1}{2}\,\text{Log}\left[1-e^{-i\,x}\right]-\frac{1}{2}\,\text{Log}\left[1-e^{i\,x}\right]$$

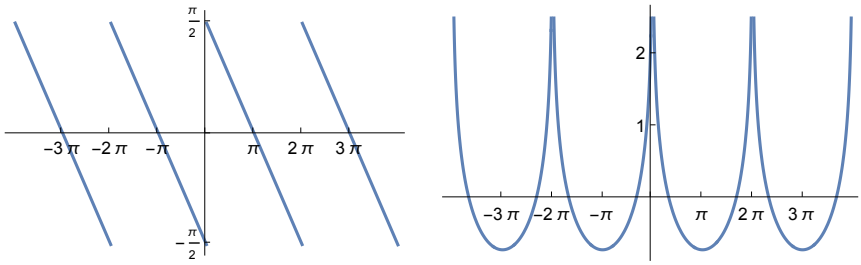
```
pb = Black;

p1 = Plot[s1, {x, -3 π, 3 π},
    Ticks → π {Range[-3, 3], Range[-1, 1] / 2}, PlotStyle → pb];
p2 = Plot[c1, {x, -3 π, 3 π}, Ticks → {π Range[-3, 3], Range[1, 5]}, PlotStyle → pb];
GraphicsRow[{p1, p2}, ImageSize → 450]
```



From the left picture it is seen that **s1** is a piecewise linear function. One can guess that in the interval (0, 2π) this function is (π - x)/2. This function must be continued periodically. This can be done with the help of the function **Floor[]**. (for this function cf. sect. 13.2).
By a simple evaluation one finds that **c1** = - log(2 | sin(x/2)|). This is a periodic function.
This gives similar graphs as above:

```
ps = Plot[N[(π - x)/2 + π Floor[x 0.5`/π]],
  {x, -4 π, 4 π}, Ticks → π {Range[-3, 3], Range[-1, 1] / 2}];
pc = Plot[-Log[2 Abs[Sin[x/2]]], {x, -4 π, 4 π},
  Ticks → {π Range[-3, 3], Range[1, 5]}];
GraphicsRow[{ps, pc}, ImageSize → 450]
```



7.1.3 Evaluation of π

The value of Ludolph's number, $\pi = 3.14159...$, can be computed by various algorithms. One may sum the well known series:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots - \ldots$$

But this series converges extremely slowly so that it is hardly usable for numerical evaluations.

```
Clear[su]
```

$$\text{su(NN\_)} = 4 \sum_{k=1}^{\text{NN}} \frac{(-1)^{k-1}}{2k-1}$$

$$\pi - 2\,(-1)^{\text{NN}}\,\text{LerchPhi}\left[-1, 1, \frac{1}{2} + \text{NN}\right]$$

```
SetPrecision[su[10^5], 10] // Timing
```

```
{3.559312, 3.14158265}
```

```
SetPrecision[%[[2]], 20]
```

```
3.1415826535897934885
```

```
SetPrecision[π, 20]
```

```
3.1415926535897932385
```

```
% - %%
```

```
9.9999999997500 × 10^-6
```

Earlier versions of Mathematia (Version 5.2) gave another formula for the above sum, which works much faster:

```
4 (π/4 - 1/4 (-1)^NN (-PolyGamma[0, 1/4 + NN/2] + PolyGamma[0, 3/4 + NN/2])) /. NN → 10^8;
SetPrecision[%, 20]
```

```
3.1415926435897932385
```

```
SetPrecision[π, 20]
```

```
3.1415926535897932385
```

```
% - %%
```

$1.0000000000 \times 10^{-8}$

From the preceeding results one sees that 10^8 terms give an accuracy of about 8 decimal places.
NSum[] uses more powerful methods to extract numerical values from the series:

```
SetPrecision[4 NSum[(-1)^(k-1)/(2 k - 1), {k, Infinity}], 19]
% - π
```

3.141592653589791340

-1.899×10^{-15}

Another method for computing π is Wallis' infinite product; but this also not very fast (s.7.2.1).
In recent years new methods have been developed for finding series, which converge faster. Practical algorithms for integer relation detection have become a staple in the emerging discipline of experimental mathematics - using modern computer technology - to explore mathematical questions.
(Computing in Science and Engineering 2 (#1,2000) 24-28, 7 (#3, 2005) 54-61).
Integer relation detection algorithms seek integers a_i such that

$$a_1x_1 + a_2x_2 + ... + a_nx_n = 0$$

is fulfilled with given real or complex numbers $x_1, x_2, ..., x_n$.
By such a method the following series for π has been found (Bailey-Borwein-Plouffe (BBP) formula)

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k}\left[-\frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} + \frac{4}{8k+1}\right]$$

```
as = Sum[1/16^k (-2/(8 k + 4) - 1/(8 k + 5) - 1/(8 k + 6) + 4/(8 k + 1)), {k, 0, 7}]
as - π // N
```

$$\frac{16\,331\,158\,360\,096\,799\,798\,177\,512\,637}{5\,198\,369\,158\,853\,231\,585\,211\,187\,200}$$

-8.20677×10^{-13}

Mathematica can calulate an arbitrary number of decimal places of π. For example:

```
SetPrecision[π, 65]
```

3.14159265358979323846264338327950288419716939937510582097494459 23

```
π =
((426 880 Sqrt[10 005]) / 13 591 409) / (HypergeometricPFQ[{1/6, 1/2, 5/6}, {1, 1}, -1/151 931 373 056 000] -
  (30 285 563 HypergeometricPFQ[{7/6, 3/2, 11/6}, {2, 2}, -1/151 931 373 056 000]) /
   1 651 969 144 908 540 723 200)
```

Mathematica currently uses the above formula, due to the Chudnovsky brothers, for calculating π to arbitrary precision.
The function **HpergeometricPFQ[]** will be explained at the end of the next section.

7.2 Products

Product[f, {i,imax}]	$\Pi_{i=1}^{imax}$ f
Product[f, {i,imin,imax}]	$\Pi_{i=imin}^{imax}$ f
Product[f, {i,imin,imax,di}]	same as above but with increment **di**
Product[f, {i,imin,imax},{j,imin,jmax}]	$\Pi_{i=imin}^{imax}\ \Pi_{i=imin}^{imax}$ f multiple product
NProduct[...]	numeric product

```
fact[n_] = Product[i, {i,n}]
```

n !

n! = Γ (n + 1)

```
? Gamma
```

Gamma[z] is the Euler gamma function $\Gamma(z)$.
Gamma[a, z] is the incomplete gamma function $\Gamma(a, z)$.
Gamma[a, z_0, z_1] is the generalized incomplete gamma function $\Gamma(a, z_0) - \Gamma(a, z_1)$. >>

The Gamma function satisfies $\Gamma(z) = \int_0^\infty t^{z-1} e^{-t}\, dt$ = (n - 1) ! for z = n = nonnegative integer.

```
fact[.5]
```

0.886227

```
0.5!
```

0.886227

```
Gamma[1.5]
```

0.886227

```
fact[n_] := Product[i, {i,n}]
```

```
fact[.5]
```

1

```
fact[2.5]
```

2

```
2.5!
```

3.32335

```
fact2[n_] = Product[i, {i,n,1,-2}]
```

$(-2)^{\text{Floor}\left[\frac{1}{2}(-1+n)\right]}\ n\ \text{Pochhammer}\left[1-\frac{n}{2}, \text{Floor}\left[\frac{1}{2}(-1+n)\right]\right]$

n!! = n (n - 2) (n - 4) ... 2 or 1 for non-negative integers. This definition of n!! by fact2[] is valid for such integers only ! The interpolation for arbitrary n is shown in Chap.13, Sect.6 .

```
fact2[5]
```

15

```
5!!
```

15

```
fact2[4]
```

8

```
4!!
```

8

```
5.5!!
```

26.5778

```
fact2[5.5]
```

28.875

Comparing the last two outputs shows that fact2[x] where x is not a non-negative integer does not agree with the interpolation used for **x!!** in *Mathematica* (§ 13.6).

```
Product[4 k (1 + k)/(1 + 2 k)^2, {k,1,Infinity} ]
```

$$\frac{\pi}{4}$$

```
Product[ 1 - 1/(k + 5)^2, {k,0,n} ]
```

$$\frac{4\,(6+n)}{5\,(5+n)}$$

```
Product[ 1 - 1/(k + 5)^(5/2), {k,0,n} ]
```

$$\prod_{k=0}^{n}\left(1-\frac{1}{(5+k)^{5/2}}\right)$$

```
NProduct[ 1 - 1/(k + 5)^(5/2), {k,0,10} ]
```

0.943002

```
NProduct[ 1 - 1/(k + 5)^(5/2), {k,0,Infinity} ]
```

0.932761

```
Product[ 1 - 1/(k + 5)^3, {k,0,n} ]
```

$$\frac{256\,\text{Cosh}\left[\frac{\sqrt{3}\,\pi}{2}\right]\,\text{Gamma}\left[\frac{13}{2}-\frac{i\sqrt{3}}{2}+n\right]\,\text{Gamma}\left[\frac{13}{2}+\frac{i\sqrt{3}}{2}+n\right]}{637\,(5+n)^3\,\pi\,\text{Gamma}[5+n]^2}$$

```
Product[ (1 - x^2/n^2), {n,Infinity}]
```

$$\frac{\text{Sin}[\pi\,x]}{\pi\,x}$$

7.2.1 Wallis' Infinite Product for π :

Lim $_{n\to\infty}$ **piu[n]** = $\pi = \frac{2}{1}\frac{2}{3}\frac{4}{3}\frac{4}{5}\frac{6}{5}\frac{6}{7}\frac{8}{7}\frac{8}{9}\cdots = \left(\frac{\prod_{k=1}^{n}2k}{\prod_{k=1}^{n}(2k-1)}\right)^2/n$

```
piu[n_] = (Product[2k,{k,n}]/Product[2k-1,{k,n}])^2/n
```

$$\frac{\pi\,\text{Gamma}[1+n]^2}{n\,\text{Gamma}\left[\frac{1}{2}+n\right]^2}$$

```
piu[10]
```

$$\frac{34\,359\,738\,368}{10\,667\,118\,605}$$

```
N[%]
```

```
3.22109
```

```
piu[100000]//N
```

```
3.1416
```

7.2.2 Hypergeometric, Confluent and Generalized Hypergeometric Functions

The **Confluent Hypergeometric Function** is defined by the following series:

$${}_1F_1(a';'b';'z) = 1 + \frac{a\,z}{b} + \frac{a\,(1+a)\,z^2}{2\,b\,(1+b)} + \frac{a\,(1+a)\,(2+a)\,z^3}{6\,b\,(1+b)\,(2+b)} + \frac{a\,(1+a)\,(2+a)\,(3+a)\,z^4}{24\,b\,(1+b)\,(2+b)\,(3+b)} + \ldots + \frac{(a)_n\,z^n}{(b)_n\,n!} + \ldots$$

$$(a)_n := a\,(a+1)\,(a+2)\ldots(a+n-1) = \text{Pochhammer}[a, n] = \Gamma(a+n)/\Gamma(a)$$

is Pochhammer's symbol.

This leads to defining the following function:

F11[a, b, z] =
1 + Sum[Product[(a+i-1)/(b+i-1) z/i, {i,k}], {k,Infinity}]

```
F11[a_, b_, z_, n_] :=
1 + Sum[ Product[(a + i - 1) / (b + i - 1) z / i, {i, k}], {k, n}]
```

The series becomes a polynomial of degree n if **a** is a negative integer; e.g. a - - 5 gives:

```
F11[-5,b,z,9]
```

$$1 - \frac{5\,z}{b} + \frac{10\,z^2}{b\,(1+b)} - \frac{10\,z^3}{b\,(1+b)\,(2+b)} + \frac{5\,z^4}{b\,(1+b)\,(2+b)\,(3+b)} - \frac{z^5}{b\,(1+b)\,(2+b)\,(3+b)\,(4+b)}$$

In *Mathematica* there is also a function programme for the confluent hypergeometric function:

```
Hypergeometric1F1[-5,b,z]
```

$$1 - \frac{5\,z}{b} + \frac{10\,z^2}{b\,(1+b)} - \frac{10\,z^3}{b\,(1+b)\,(2+b)} + \frac{5\,z^4}{b\,(1+b)\,(2+b)\,(3+b)} - \frac{z^5}{b\,(1+b)\,(2+b)\,(3+b)\,(4+b)}$$

The genuine **Hypergeometric Function** as defined by Gauss is:

$${}_2F_1(a, b; c; z) = 1 + \frac{a\,b\,z}{c} + \frac{a\,(1+a)\,b\,(1+b)\,z^2}{2\,c\,(1+c)} + \frac{a\,(1+a)\,(2+a)\,b\,(1+b)\,(2+b)\,z^3}{6\,c\,(1+c)\,(2+c)} + \ldots + \frac{(a)_n\,(c)_n\,z^n}{(c)_n\,n!} + \ldots$$

The **generalized Hypergeometric Function** is:

$$ {}_pF_q\left(a_1, a_2, \ldots a_p\,';\,' b_1, b_2, \ldots b_q\,';\,' z\right) = 1 + \sum_{n=1}^{\infty} \frac{(a_1)_n\,(a_2)_n \ldots \left(a_p\right)_n\, z^n}{(b_1)_n\,(b_2)_n \ldots \left(b_q\right)_n\, n!} $$

In *Mathematica* it is written as:

```
HypergeometricPFQ[{a1, a2, ... ap},{b1, b2, .. bq},z]
```

7.3 General Loops

7.3.1 Unconditional Loops

Do[f, {n}]	Perform task f n times
Do[f, {i,imax}]	Perform task f from i = 1 to i = imax
Do[f, {i,imin,imax}]	Perform task f from i = imin to i = imax
Do[f, {i,imin,imax,di}]	Perform task f from i = imin to i = imax incrementing i by di
Do[f, {i,imin,imax},{j,jmin,jmax}]	double loop

```
Do[ Print[i^2], {i,4} ]
```

```
1
4
9
16
```

```
Do[ Print[{i, i^3}], {i,3,12,4}]
```

```
{3, 27}
{7, 343}
{11, 1331}
```

```
Do[ Print[{i,j}], {i,4}, {j,i-1} ]
```

```
{2, 1}
{3, 1}
{3, 2}
{4, 1}
{4, 2}
{4, 3}
```

The following command gives a finite continued fraction (= Kettenbruch).

```
t = x; Do[t = 1/(1 + t),{3}];t
```

$$ \frac{1}{1+\frac{1}{1+\frac{1}{1+x}}} $$

```
% /. x -> 5
```

$$\frac{7}{13}$$

7.3.2 Conditional Loops

Do-Loops may be stopped, interrupted or branched by conditions; the conditions can be expressed with the help of an **If[]** command. The action to be taken may be specified by one of the following commands:

Break[]	exit the nearest enclosing loop
Continue[]	go to the next step in the current loop
Return[*expr*]	return the value *expr*, exiting all procedures and loops in a function
Goto[*name*]	go to the element **Label[*name*]** in the current procedure

```
Do[ t = k^2; Print[t]; If[t > 10, Break[] ], {k,7} ]
```

```
1
4
9
16
```

```
Do[ t = k^2;  If[t > 10,Break[],Print[t]], {k,7} ]
```

```
1
4
9
```

```
Do[ t = k^2;  If[t > 25, Return[t] ], {k,9} ]
```

```
36
```

```
Do[ t = k^2; Print[{k, t}];
    If[ t < 17, Continue[], Break[] ], {k,9} ]
```

```
{1, 1}
{2, 4}
{3, 9}
{4, 16}
{5, 25}
```

There are several other commands for (unconditional and conditional) loops:

Nest, FixedPoint, For, While.

7.3.3 FixedPoint

FixedPoint[f, expr]	starts with **expr**, then applies **f** repeatedly until the result no longer changes.
FixedPoint[f, expr, n]	stops after at most **n** steps.

7.3.3.1 A simple example for FixedPoint

```
FixedPoint[Cos[#] &, 1, 7]
```

```
Cos[Cos[Cos[Cos[Cos[Cos[Cos[1]]]]]]]
```

```
N[%]
```

```
0.722102
```

```
FixedPoint[Cos[#] &, 1, 35] // N
```

```
0.739085
```

```
FixedPoint[Cos[#] &, 1.]
```

```
0.739085
```

7.3.3.2 Implementing Newton's method

Newtons method computes an approximate zero of f(x) by the following iteration (s. § 9.5) :

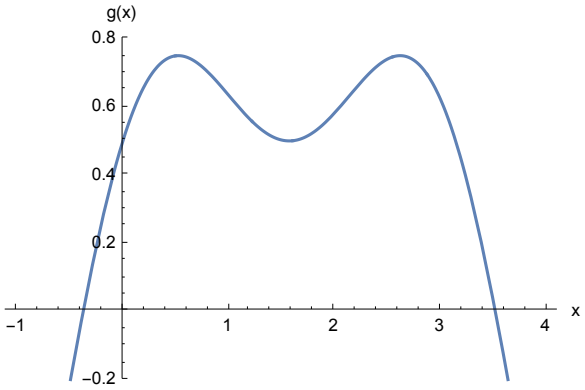
$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

This formula is implemented in the following way:

```
myFindRoot[f_, init_] := FixedPoint[#1 - f[#1]/f'[#1] &, init];
```

```
Clear[x, g]; g[x_] = Sin[x] + Cos[2 x] / 2;
```

```
Plot[g[t], {t, -1, 4}, AxesLabel → {"x", "g(x)"},
 PlotRange → {-0.2, 0.8}, ImageSize → 300]
```




```
myFindRoot[g, 3.]
```

```
3.51633
```

7.4 Conditional Branching

There are several commands for conditional branching. The simplest is:

```
If[test, then, else]

If[test, then]

If[test,then, else, unknown]
```

If the answer to test is **True**, the statement(s) in position **then** are executed; if it is **False**, the statement(s) in position **else** are executed. The **If[]** with four arguments provides for the case that the answer to test may be undecided; then **unknown** is executed.
Now this is used to define the Heaviside unit step function:

```
theta[x_]  =  If[x > 0, 1, 0]
```

```
If[x > 0, 1, 0]
```

```
theta[1]
```

1

```
theta[0]
```

0

```
theta[-1]
```

0

Note the order of the symbols for "ge" , viz. >= and "le", viz. <= .

```
thetag[x_] = If[x >= 0, 1, 0]
```

```
If[x ≥ 0, 1, 0]
```

```
thetag[1]
```

1

```
thetag[0]
```

1

```
thetag[-1]
```

0

```
thetal[x_] = If[x <= 0, 0, 1]
```

```
If[x ≤ 0, 0, 1]
```

```
thetal[1]
```

1

```
thetal[0]
```

0

```
thetal[-1]
```

0

The Kronecker symbol δ_{nk} may be defined as:

```
Remove[n,k]
krodel[n_,k_] = If[ k == n, 1, 0]
```

```
If[k == n, 1, 0]
```

```
krodel[3,4]
```

0

```
krodel[3,3]
```

1

```
krodel[a,a]
```

1

```
Remove[n,k]
krodel[n_,k_] = If[ k != n, 0, 1]
```

```
If[k ≠ n, 0, 1]
```

```
krodel[3,4]
```

0

```
krodel[a,a]
```

1

```
k = 1;
If[k == 1, Print["yes"]]
```

yes

```
If[k != 1, Print["yes"]]
```

A more general type of command is:

Which[test1, value1, test2, value2, ...]

Tests are applied in the order given starting from left to right. The value belonging to the test which comes out **True** is given. A last condition "**True**" may cover the cases remaining after all preceeding tests have met "**False**".

```
theta[x_] = Which[x > 0, 1, x == 0, 1/2, x < 0, 0]
```

$$\text{Which}\left[x > 0,\ 1,\ x == 0,\ \frac{1}{2},\ x < 0,\ 0\right]$$

```
theta[1]
```

1

```
theta[0]
```

$$\frac{1}{2}$$

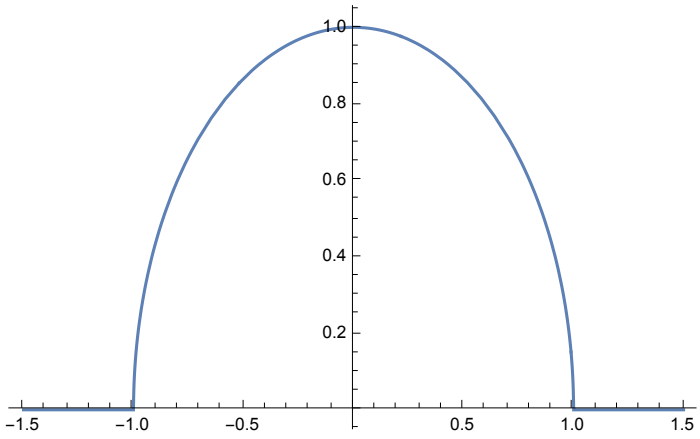
```
theta[-1]
```

0

```
Clear[x]
h[x_] = Which[x^2 < 1, Sqrt[1 - x^2], True, 0]
```

$$\text{Which}\left[x^2 < 1,\ \sqrt{1 - x^2},\ \text{True},\ 0\right]$$

```
Plot[h[x], {x, -1.5`, 1.5`}]
```



Note that logical equality must be expressed by " == ". Otherwise an error will occur:

```
theta[x_] = Which[x > 0, 1, x = 0, 1/2, x < 0, 0]
```

$$\text{Which}\left[x > 0,\ 1,\ x = 0,\ \frac{1}{2},\ x < 0,\ 0\right]$$

```
theta[1]
```

```
1
```

```
theta[0]
```

```
Set::setraw : Cannot assign to raw object 0. >>
```

```
Which[0, 1/2, 0 < 0, 0]
```

Set::setraw : Cannot assign to raw object 0 . >>

7.4.1 A Test Comprising Several Conditions

A test may comprise several conditions. These must be linked either by **&&** or by **||** :

&& (= ∧) Result of test is True if all conditions are met.

|| (= ∨) Result of test is True if at least one conditions is met.

```
theta2[x_,y_] = If[x > 0  &&  y > 0, 1, 0]
```

```
If[x > 0 && y > 0, 1, 0]
```

```
theta2[1,1]
```

```
1
```

```
theta2[-1,1]
```

```
0
```

```
theta2[1,-1]
```

```
0
```

```
theta2[-1,-1]
```

```
0
```

```
theta2[x_,y_] = If[x > 0  ||  y > 0, 1, 0]
```

```
If[x > 0 || y > 0, 1, 0]
```

```
theta2[1,1]
```

```
1
```

```
theta2[-1,1]
```

```
1
```

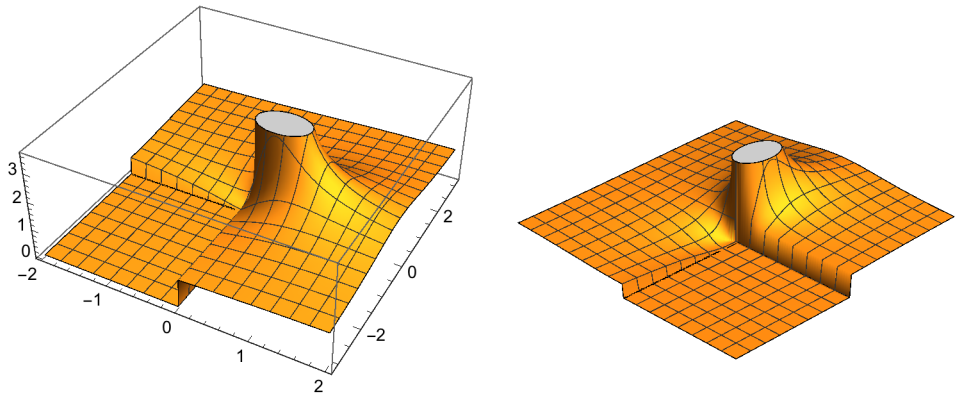
```
theta2[1,-1]
```

```
1
```

```
theta2[-1,-1]
```

```
0
```

```
Clear[f,g,x,y]
g[x_,y_] = Abs[ Exp[ (x + I y)^-1 ] ];
f[x_,y_] = Which[ x < 0 && y < 0  (*third quadrant *), 0,
                  True (* all other quadrants *), g[x,y] ];
re = Plot3D[f[x, y], {x, -2, 2}, {y, -π, π}, PlotPoints → 50]; li = Show[re, PlotRange
GraphicsRow[{re, li}, ImageSize → 500]
```




```
f1[x_] := Which[x < 0, 0, x < 1, 1, x < 2, 2]

SetAttributes[f1, Listable]

f1[{-0.5, 0., 0.75, 1., 1.5, 2, 3}]
```

```
{0, 1, 1, 2, 2, Null, Null}
```

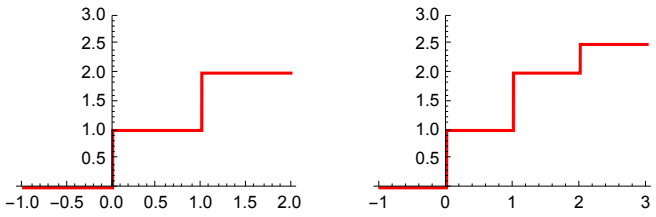
```
f2[x_] := Which[x < 0, 0, x < 1, 1, x < 2, 2, True, 2.5]

SetAttributes[f2, Listable]

f2[{-0.5, 0., 0.75, 1., 1.5, 2, 3}]
```

```
{0, 1, 1, 2, 2, 2.5, 2.5}
```

```
p1 = Plot[f1[x], {x, -1, 3}, PlotStyle → Hue[0], PlotRange → {0, 3}];
p2 = Plot[f2[x], {x, -1, 3}, PlotStyle → Hue[0], PlotRange → {0, 3}];
GraphicsRow[{p1, p2}]
```



7.4.2 Functions for Testing Properties of Expressions

Besides the logical operators

```
<    <=    ==    !=      >    >=    &&    ||
```

there are also functions which may be used for expressing conditions by checking properties of expressions. They end with the letter **Q** (= Question).

NumberQ[*expr*]	number
IntegerQ[*expr*]	integer
EvenQ[*expr*]	even number
OddQ[*expr*]	odd number
PrimeQ[*expr*]	prime number
VectorQ[*expr*]	a list representing a vector
MatrixQ[*expr*]	a list representing a matrix
SameQ[*expr1*,*expr2*]	test whether expressions *expr1* and *expr2* are the same

```
NumberQ[x]
```

```
False
```

```
NumberQ[Pi]
```

```
False
```

```
NumberQ[3.14]
```

```
True
```

```
IntegerQ[3.14]
```

```
False
```

```
PrimeQ[5679]
```

```
False
```

```
PrimeQ[23]
```

```
True
```

```
fact[x_] = If[IntegerQ[x], Product[k,{k,x}], Gamma[x+1] ];
```

```
fact[x]
```

```
Gamma[1 + x]
```

```
fact[5]
```

```
120
```

```
fact[4.5]
```

```
52.3428
```

```
4.5!
```

```
52.3428
```

```
fact2[x_] :=
If[IntegerQ[x],Print[Product[k,{k,x,1,-2}]], Print[x!!] ];
```

```
fact2[4.5]
```

```
10.8318
```

```
fact2[5]
```

```
15
```

Defining the same function without using delayed definition does not work properly.

```
fact2[x_] =
If[IntegerQ[x], Print[Product[k,{k,x,1,-2}]], Print[x!!] ];
```

```
x!!
```

```
fact2[4.5]
```

```
fact2[5]
```

```
fact2[x_] = If[IntegerQ[x],
Return[Product[k,{k,x,1,-2}]], Return[x!!] ];
```

```
Return[x!!]
```

For **Return[]** cf. Chp. 7.4 .

```
Clear[a, b, c, c, s, ma]
ma = {{a + b, 0, c - 1, 0}, {0, d 5, 0, 12 b}, {s + 1, 0, a - d, 0}, {0, 1, 0, 4}};
MatrixForm[ma]
```

$$\begin{pmatrix} a+b & 0 & -1+c & 0 \\ 0 & 5\,d & 0 & 12\,b \\ 1+s & 0 & a-d & 0 \\ 0 & 1 & 0 & 4 \end{pmatrix}$$

Below the structure of the above matrix is to be displayed: All non-zero matrix elements are denoted by a letter x.

```
MatrixForm[
 Table[If[SameQ[ma[[i, j]], 0], 0, x], {i, Length[ma]}, {j, Length[ma]}]]
```

$$\begin{pmatrix} x & 0 & x & 0 \\ 0 & x & 0 & x \\ x & 0 & x & 0 \\ 0 & x & 0 & x \end{pmatrix}$$

7.5 Exercises

7.1 Compute the sum of the following series :
a) $\sum_{k=0}^{\infty} 1/3^k$; b) $\sum_{k=2}^{\infty} (-1)^k/\ln(k)$; c) $\sum_{k=1}^{\infty} \sin(n\,x)/n$, insert $-\pi/2$ and 1.6 ;
d) 1 + 1/2 - 1/3 - 1/4 + 1/5 + 1/6 - 1/7 - ... + ... + ...

7.2 The periodic saw tooth curve is given by the following Fourier series:
$x \sim [-\sin(x) + \sin(2x)/2 - \sin(3x)/3 + \text{-} \ldots]$, $-\pi < x < \pi$.
Plot the exact saw tooth curve, results of the partial sums, Gibb's phenomenon and Fejer sums.

7.3 Compute the sum of the following Fourier series and find the intervall in which the series represents the polynomial:
a) $\sum_{n=1}^{\infty} \cos(nx)/n^2 = \pi^2/6 - x\,\pi/2 + x^2/4$;
b) $\sum_{n=1}^{\infty} \sin(n\,x)/n^3 = x\,\pi^2/6 - x^2\,\pi/4 + x^3/12$.

7.4 Compute and print $\prod_{k=1}^{n} (1 - e^{2\,\pi\,i\,k/n})$ for n = 1,2,....,9.

7.5 Compute the doubly infinite product $\prod_{n=-\infty}^{\infty} (1 - z/(n - 1/2))$.

7.6 Repeat ex. 6.7. The lists for the options generating ticks and gridlines should be generated by a combination of the command Table and commands for branchings.

7.7 Repeat ex. 6.9. The curve should be generated with one Plot command and a branching command.

7.8 In the list **d** given below there are relative maxima. Find these and give these together with their position in the list d.

```
d = {1, 2, 3, 6, 11, 7, 6, 4, 6, 8, 11, 17, 12, 10, 8, 6, 3, 3};
```