Stirling interpolation formula.

Polynomial of order $m = 2n$ (Antiqu.)

↑ even, in other words.

$$F_{2n}(x) = f_k + u\,\mu\delta f_k + \frac{u^2}{2!}\delta^2 f_k + \dots$$

$$= f_k + \sum_{l=1}^{n}{}' \binom{u+l-1}{2l-1} \left\{ \mu\delta^{2l-1} f_k + \frac{u}{2l}\delta^{2l} f_k \right\}$$

$$+ O(\Delta x^{2n+1})$$

$$u = \frac{x - x_k}{\Delta x}$$

Example:

$n = 1$ $\quad (m = 2)$

$$F_2(x) = f_k + \frac{\mu \delta f_k}{\Delta x}(x - x_k) + \frac{1}{2}\frac{\delta^2 f_k}{\Delta x^2}(x - x_k)^2 + O(\Delta x^3)$$

This is the parabolic Shilnij formula.

Diffence quotients:

$$\frac{d}{d\mu}\binom{\mu}{\ell} = \binom{\mu}{\ell} \sum_{i=0}^{\ell-1}{}' \frac{1}{\mu - i}$$

$$\frac{d^2}{d\mu^2}\binom{\mu}{\ell} = \begin{cases} 0 & \ell = 1 \\ \binom{\mu}{\ell} \sum_{i=0}^{\ell-1}{}' \sum_{j=0}^{\ell-1}{}' \frac{1}{(\mu - i)(\mu - j)}, & \ell \geq 2. \end{cases}$$

DNGF - formulas

Difference - Newton - Gregory - formula

$$F_m'(x) = \frac{1}{\Delta x} \sum_{l=1}^{m}{}' \Delta^l f_k \binom{u}{l} \sum_{i=0}^{l-1} \frac{1}{u-i} + O(\Delta x^m)$$

$$F_m''(x) = \frac{1}{\Delta x^2} \sum_{l=2}^{m}{}' \Delta^l f_k \binom{u}{l} \sum_{i=0}^{l-1}{}' \sum_{j=0}^{l-1}{}' \frac{1}{(u-i)(u-j)} + O(\Delta x^{m-1})$$

Now, we set $x = x_k$:

$$F_m'(x_k) = \frac{1}{\Delta x} \sum_{l=1}^{m}{}' (-1)^{l-1} \frac{\Delta^l f_k}{l} + O(\Delta x^m)$$

$$F_m''(x_k) = \frac{1}{\Delta x^2} \sum_{l=2}^{m}{}' (-1)^{l+1} \frac{\Delta^l f_k}{l} \sum_{i=1}^{l-1}{}' \frac{1}{i} + O(\Delta x^{m-1})$$

Example:

Setting $m = 2$:

$$F_2'(x_k) = \frac{1}{\Delta x}\left(\Delta f_k - \frac{\Delta^2 f_k}{2}\right) + O(\Delta x^2)$$

$$= \frac{1}{\Delta x}\left(-\frac{1}{2} f_{k+2} + 2 f_{k+1} - \frac{3}{2} f_k\right) + O(\Delta x^2)$$

DNGB

Example:

$$F_2'(x_k) = \frac{1}{\Delta x}\left(\nabla f_k - \frac{\nabla^2 f_k}{2}\right) + O(\Delta x^2)$$

$$= \frac{1}{\Delta x}\left(\frac{3}{2} f_k - 2 f_{k-1} + \frac{1}{2} f_{k-2}\right)$$

$$+ O(\Delta x^2).$$

DST - Differential Shifting formula

$$F'_{2m}(x_k) = \frac{1}{\Delta x}\left(\mu\delta f_k - \frac{1}{6}\mu\delta^3 f_k + \frac{1}{30}\mu\delta^5 f_k - \frac{1}{140}\mu\delta^7 f_k + \dots\right) + O(\Delta x^{2m})$$

$$F''_{2m}(x_k) = \frac{1}{\Delta x^2}\left(\delta^2 f_k - \frac{1}{12}\delta^4 f_k + \frac{1}{90}\delta^6 f_k - \frac{1}{560}\delta^8 f_k + \dots\right) + O(\Delta x^{2m})$$

!

In the DNGF and DNGB schemes, the error increases for each new order of derivation. This does not happen in the DST - scheme!

Example:

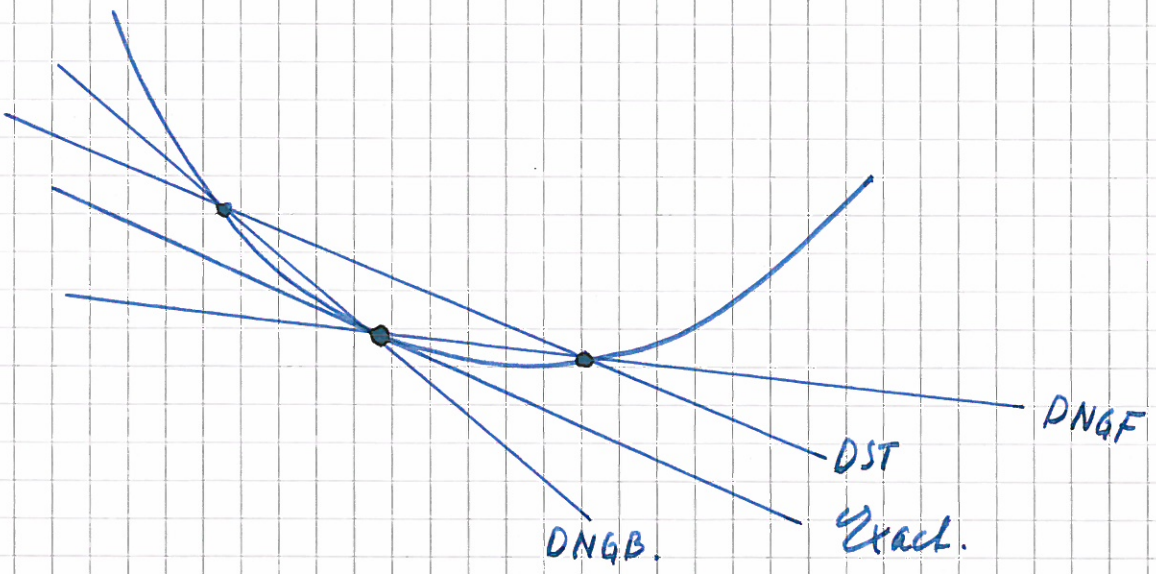
$$F_2'(x_k) = \frac{1}{\Delta x}\left(\mu \delta f_k\right) + O(\Delta x^2)$$

$$= \frac{1}{2\Delta x}\left(f_{k+1} - f_{k-1}\right) + O(\Delta x^2)$$

$$F_2''(x_k) = \frac{\delta^2 f_k}{\Delta x^2} + O(\Delta x^2)$$

$$= \frac{1}{\Delta x^2}\left(f_{k+1} - 2f_k + f_{k-1}\right) + O(\Delta x^2).$$

Why does this work so well? Consider the following graphical representation.



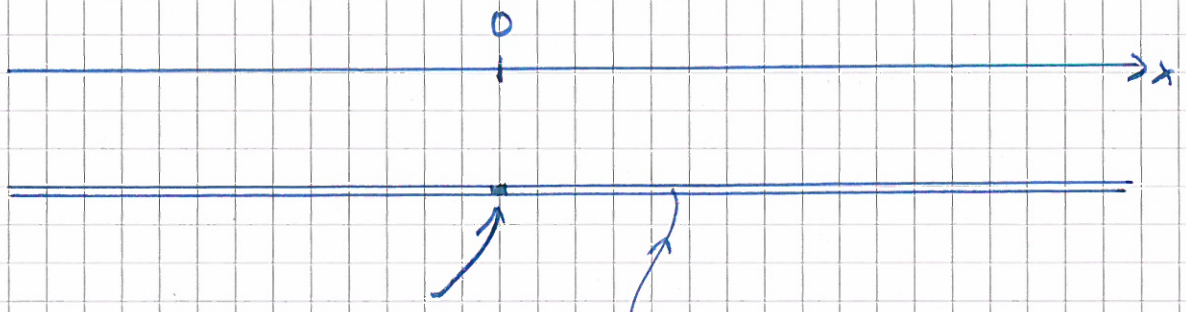
3. Stochastic Processes.

Stochastics is the opposite of Statistics.

Statistics: Generate distributions from data.

Stochastics: Generate data from distributions.

Example: Diffusion.



Ink thin, waterfilled tube.

How does the ink spread in time?

Ink concentration: $P(x,t)$

Diffusion equation: $\boxed{\frac{\partial P}{\partial t} = \frac{D}{2} \frac{\partial^2 P}{\partial x^2}}$

Initial condition: $P(x,0) = \delta(x)$

Solution: $P(x,t) = \frac{e^{-x^2/2Dt}}{\sqrt{\pi D t}}$

Incidentally, diffusion coefficient ~ $D = 0.1 \cdot \frac{mm^2}{s}$.
for water molecules in water.

The diffusion equation is an example of a <u>Fokker-Planck equation.</u>

For every Fokker - Planck equation, there is a corresponding Langevin equation.

$$\frac{\partial P}{\partial t} = \frac{D}{2} \frac{\partial^2 P}{\partial x^2} \iff \boxed{\dot{x} = \eta(t)}$$

Noise term.

$$\begin{cases} \langle \eta \rangle = 0 \\ \langle \eta(t)\,\eta(t') \rangle = D\,\delta(t-t') \end{cases}$$

Uncorrelated white noise.

Integrating the Langevin equation:

$$x(t) = \underbrace{x_0}_{=0} + \int_0^t \eta(t')\,dt' = \int_0^t \eta(t')\,dt'$$

We set $x_0 = 0$

$$\langle x(t) \rangle = \left\langle \int_0^t \eta(t')\,dt' \right\rangle$$

$$= \int_0^t \langle \eta(t') \rangle\,dt' = 0$$

$$\langle x^2(t) \rangle = \left\langle \int_0^t \eta(t')\,dt' \int_0^t \eta(t'')\,dt'' \right\rangle$$

$$= \int_0^t dt' \int_0^t dt'' \langle \eta(t')\,\eta(t'') \rangle$$

$$= D \int_0^t dt' \int_0^t dt'' \quad \delta(t'-t'') = D \int_0^t dt' = Dt$$

From the solution of the diffusion equation

$$P(x,t) = \frac{e^{-x^2/2Dt}}{\sqrt{2\pi D t}}$$

$$\langle x \rangle = \int_{-\infty}^{+\infty} dx \; x\, P(x,t) = 0$$

$$\langle x^2 \rangle = \int_{-\infty}^{+\infty} dx \; x^2 P(x,t) = Dt$$

All moments of x calculated with the Langevin equation and the corresponding Fokker-Planck equation give the same result.

$\Rightarrow$ Langevin equation gives a stochastic realization of the corresponding Fokker-Planck equation.

Random numbers.

Benford's law

The first digit of numbers occuring naturally in life follow the law

$$\log_{10}\left(1+\frac{1}{d}\right).$$

$d=1$	$\log_{10}\left(1+\frac{1}{1}\right)$	$= 30.1\%$
$d=2$	$\log_{10}\left(1+\frac{1}{2}\right)$	$= 17.6\%$
$d=3$	$\log_{10}\left(1+\frac{1}{3}\right)$	$= 12.5\%$
$d=4$	$\log_{10}\left(1+\frac{1}{4}\right)$	$= 9.7\%$
$d=5$	$\log_{10}\left(1+\frac{1}{5}\right)$	$= 7.9\%$
$d=6$	$\log_{10}\left(1+\frac{1}{6}\right)$	$= 6.7\%$
$d=7$	$\log_{10}\left(1+\frac{1}{7}\right)$	$= 5.8\%$
$d=8$	$\log_{10}\left(1+\frac{1}{8}\right)$	$= 5.1\%$
$d=9$	$\log_{10}\left(1+\frac{1}{9}\right)$	$= 4.6\%$

Why!

Probability density without internal scale: $\frac{dx}{x}$

$$\int_d^{d+1} \frac{dx}{x} = \ln(d+1) - \ln d = \ln\left(1+\frac{1}{d}\right)$$

Normalizing:

$$\int_1^{10} \frac{dx}{x} = \ln 10$$

Probability: $\frac{\ln(1+\frac{1}{d})}{\ln 10} = \log_{10}(1+\frac{1}{d})$

How to generate random numbers that can be trusted.

$$ran \rightarrow r \quad ; \quad 0 < r < 1.$$

Random number generators should be:

1. Random
2. Reproducible
3. Portable
4. Efficient.

There is no universal random number generator. Always test your random number generator in the application you have in mind.

More complex random number generators $\not\Rightarrow$ better generators.

Often, the opposite is the case.

Linear congruence random number generator

$$I_{j+1} = (a I_j + c) \cdot \text{mod}. m$$

↑ Modulus.

(Modulus: $j \cdot \text{mod} \cdot m = j - \underbrace{\left[\frac{j}{m}\right]}_{\text{Integer division}} \cdot m$

Example: $7 \cdot \text{mod} \cdot 5 = 7 - \left[\frac{7}{5}\right] \cdot 5 = 7 - 1 \cdot 5 = 2$)

$\left[\frac{7}{5}\right] = 1$

Period $\leq m$, hopefully $\simeq m$.

m is usually large; $m = 2^{32} \simeq 4.3 \cdot 10^9$.

Generators produce sequences of random numbers. Group them.

$\vec{R}_i = \underbrace{(r_1, r_2, \ldots, r_k)_i}_{\text{sequence points in a } k\text{-dimensional box}}$

i: ith sequence

Plot $\vec{R}_1, \vec{R}_2, \ldots, \vec{R}_N$.

The points will fall on $(k-1)$-dimensional planes.

The number of planes will be $m^{1/k}$ or less.

The better the generator, the more planes.

Lewis, Goodman and Miller

"Minimal Standard Generator"

$$I_{j+1} = (16807 \cdot I_j) \bmod 2147483647$$

$$= 2^{31} - 1.$$

"Magic number"

Produces random numbers in the range

$$-(2^{31}-1) \leq I_{j+1} < (2^{31}-1)$$

on a 32-bit machine.

(Why can multiplication of two positive numbers give a negative number?)

Very well tested and can usually be trusted.

Bits of higher order (more significant - further to the left) can be trusted more than bits of low order.

To generate random numbers between 1 and 10, use the following method.

$$j = 1 + \text{int}(10 * r).$$

$0 < r < 1$.

This would be significantly more:

$$j = 1 + (\text{int}(1000 * r)) \bmod 10.$$

Where do the following to generate 10 random bits:

```
do  i = 1, 10
k = and (1, ishft (I, i))
enddo
```

↑ random number.

Kirkpatrick - Stoll generator R250.

Initially:

250 integers containing random bits.

N(1), ..., N(250).

Based on the XOR logical operation:

XOR	1	0
1	0	1
0	1	0

N(251) = XOR (N(1), N(178))

↑ A new random number.

Generally:

N(k) = XOR (N(k-250), N(k-103))

↑ magic numbers.

To generate the first 250 random numbers $N(1), .., N(250)$:

```
do k = 1, 250
ICI = 0
do I = 1, 32
ICI = ISHIFT (ICI, 1)
IBM = IBM * 16807
IF (IBM .LT. 0) ICI = ICI + 1
enddo
N(k) = ICI .
enddo
```

Hence, use 16807 to generate each bit in $N(1), .., N(250)$.

Jake implementation of R250:

N(and (255, k))

Reads the 8 last bits of k.

= XOR (N(and (255, k-250)),

N(and (255, k-103))

Now, k may proceed to ∞, but the dimension of N is still only 256:

N(0), .., N(255).

Which generator is best of 16807 and R250?

Before, R250 was considered best, but in 1992, Landau's group at the University of Georgia found a 1%

error in the Ising model when using the R250. 1680 worked well.

However, Dietrich Stauffer at the University of Cologne has had problems with 1680 in connection with the Ising model.

Some hints:

- ✓ Initialize 1680 with odd numbers.
- ✓ Use odd lattice sizes (L = 31, rather than L = 32).
- ✓ Generate an extra random # from time to time.

Another "magic" number:

65539.

Use this to test 16807 (by substituting 16807 by 65539).

Even use 65539 for production runs.

Generating numbers x that follow a probability distribution $p(x)$.

Cumulative probability

$P(x)$

↑ Probability to find a value $x' \leq x$.

$P(-\infty) = 0$

$P(\infty) = 1$.