

conventions and some tests for FFT\_xl.xls

```
> restart;  
> Digits:=16;
```

Digits := 16

### conventions for summations

```
> 'fft(k)=alpha*Sum( (x[j+1]+I*y[j+1])*exp(-2*Pi*I*j*k/N) ,j=0..N-1)'; `with alpha=1`;
```

$$\text{fft}(k) = \alpha \left( \sum_{j=0}^{N-1} (x_{j+1} + y_{j+1} I) e^{\left(\frac{-2i\pi jk}{N}\right)} \right)$$

with alpha=1

```
> 'inverse_fft(k)=beta*Sum( (x[j+1]+I*y[j+1])*exp(2*Pi*I*j*k/N) ,j=0..N-1)'; `with beta=1/N`;
```

$$\text{inverse\_fft}(k) = \beta \left( \sum_{j=0}^{N-1} (x_{j+1} + y_{j+1} I) e^{\left(\frac{2i\pi jk}{N}\right)} \right)$$

with beta=1/N

and for VBA coding use

```
> 'eval((x[j+1]+y[j+1]*I)*exp(-2*I*Pi*j*k/theN),j=j-1)': '%'= simplify(evalc(%));
```

$$(x_{j+1} + y_{j+1} I) e^{\left(\frac{-2i\pi jk}{\text{theN}}\right)} \Big|_{j=j-1} = x_j \cos\left(\frac{2\pi(j-1)k}{\text{theN}}\right) + y_j \sin\left(\frac{2\pi(j-1)k}{\text{theN}}\right) + y_j \cos\left(\frac{2\pi(j-1)k}{\text{theN}}\right) I - x_j \sin\left(\frac{2\pi(j-1)k}{\text{theN}}\right) I$$

```
> 'eval((x[j+1]+I*y[j+1])*exp(2*Pi*I*j*k/theN),j=j-1)': '%'= simplify(evalc(%));
```

$$(x_{j+1} + y_{j+1} I) e^{\left(\frac{2i\pi jk}{\text{theN}}\right)} \Big|_{j=j-1} = x_j \cos\left(\frac{2\pi(j-1)k}{\text{theN}}\right) - y_j \sin\left(\frac{2\pi(j-1)k}{\text{theN}}\right) + y_j \cos\left(\frac{2\pi(j-1)k}{\text{theN}}\right) I + x_j \sin\left(\frac{2\pi(j-1)k}{\text{theN}}\right) I$$

```
> n:=5;N:=2^n;
```

n := 5

N := 32

### test data for the arrays x, y

```
> [seq(i,i=0..N-1)]: nops(%);  
x:=convert(%,array);
```

32

```
x := [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31]
```

```
> [seq(-(N-i),i=0..N-1)]: nops(%);  
y:=convert(%,array);
```

32

```
y := [-32, -31, -30, -29, -28, -27, -26, -25, -24, -23, -22, -21, -20, -19, -18, -17, -16, -15, -14, -13, -12, -11, -10, -9, -8, -7, -6, -5, -4, -3, -2, -1]
```

FFT in Maple 8 over-writes the arrays, so save in a copy first

```
> xOrig:=copy(x): yOrig:=copy(y):
```

### test results

```
> x:=copy(xOrig): y:=copy(yOrig): # work with the copy  
FFT(n,x,y): # apply FFT and print result
```

```
printf (" FFT of z = x + y*I:");  
for k from 1 to N do  
printf ("%2d: %+3.8f %+3.8f \n",k,x[k],y[k]);  
end do;
```

```
FFT of z = x + y*I:  
1: +496.00000000 -528.00000000  
2: -178.45072620 +146.45072620  
3: -96.43743187 +64.43743187  
4: -68.74493134 +36.74493134  
5: -54.62741700 +22.62741700  
6: -45.93389459 +13.93389459  
7: -39.94569220 +7.94569220  
8: -35.49605641 +3.49605641  
9: -32.00000000 +0.00000000  
10: -29.13086065 -2.86913935  
11: -26.69085821 -5.30914179  
12: -24.55217818 -7.44782182  
13: -22.62741700 -9.37258300  
14: -20.85354694 -11.14645306  
15: -19.18259788 -12.81740212  
16: -17.57586245 -14.42413755  
17: -16.00000000 -16.00000000  
18: -14.42413755 -17.57586245  
19: -12.81740212 -19.18259788  
20: -11.14645306 -20.85354694  
21: -9.37258300 -22.62741700  
22: -7.44782182 -24.55217818  
23: -5.30914179 -26.69085821  
24: -2.86913935 -29.13086065  
25: +0.00000000 -32.00000000  
26: +3.49605641 -35.49605641  
27: +7.94569220 -39.94569220  
28: +13.93389459 -45.93389459  
29: +22.62741700 -54.62741700
```

```

30: +36.74493134 -68.74493134
31: +64.43743187 -96.43743187
32: +146.45072620 -178.45072620
> x:=copy(xOrig): y:=copy(yOrig): # work with the copy
  iFFT(n,x,y): # apply FFT and print result

printf (" inverse FFT of z = x + y*I:");
for k from 1 to N do
  printf ("%2d: %+3.8f %+3.8f \n",k,x[k],y[k]);
end do;

inverse FFT of z = x + y*I:
1: +15.50000000 -16.50000000
2: +4.57658519 -5.57658519
3: +2.01366975 -3.01366975
4: +1.14827910 -2.14827910
5: +0.70710678 -1.70710678
6: +0.43543421 -1.43543421
7: +0.24830288 -1.24830288
8: +0.10925176 -1.10925176
9: +0.00000000 -1.00000000
10: -0.08966060 -0.91033940
11: -0.16591068 -0.83408932
12: -0.23274443 -0.76725557
13: -0.29289322 -0.70710678
14: -0.34832666 -0.65167334
15: -0.40054382 -0.59945618
16: -0.45075430 -0.54924570
17: -0.50000000 -0.50000000
18: -0.54924570 -0.45075430
19: -0.59945618 -0.40054382
20: -0.65167334 -0.34832666
21: -0.70710678 -0.29289322
22: -0.76725557 -0.23274443
23: -0.83408932 -0.16591068
24: -0.91033940 -0.08966060
25: -1.00000000 +0.00000000
26: -1.10925176 +0.10925176
27: -1.24830288 +0.24830288
28: -1.43543421 +0.43543421
29: -1.70710678 +0.70710678
30: -2.14827910 +1.14827910
31: -3.01366975 +2.01366975
32: -5.57658519 +4.57658519

```