API Collection

In-Place Functions for 2D Complex FFT

Perform fast Fourier transforms in place on 2D complex data.

Overview

The functions in this group use the following operation for a complex-to-complex transform:

```
N0 = 1 << Log2N0;
N1 = 1 << Log2N1;

if (IC1 == 0) IC1 = IC0*N0;

scale = 0 < Direction ? 1 : 1. / (N1*N0);

// Define a complex matrix, h:
for (j1 = 0; j1 < N1; ++j1)
for (j0 = 0; j0 < N0; ++j0)
    h[j1][j0] = C->realp[j1*IC1 + j0*IC0]
           + i * C->imagp[j1*IC1 + j0*IC0];

// Perform Discrete Fourier Transform.
for (k1 = 0; k1 < N1; ++k1)
for (k0 = 0; k0 < N0; ++k0)
    H[k1][k0] = scale * sum(sum(h[j1][j0]
        * e**(-Direction*2*pi*i*j0*k0/N0), 0 <= j0 < N0)
        * e**(-Direction*2*pi*i*j1*k1/N1), 0 <= j1 < N1);

// Store result.
for (k1 = 0; k1 < N1; ++k1)
for (k0 = 0; k0 < N0; ++k0)
{
    C->realp[k1*IC1 + k0*IC0] = Re(H[k1][k0]);
    C->imagp[k1*IC1 + k0*IC0] = Im(H[k1][k0]);
}
```

The temporary buffer versions perform the same operation but use a temporary buffer for improved performance.

Topics

In-Place FFT Functions

`vDSP_fft2d_zip`

Computes a 2D forward or inverse in-place, single-precision complex FFT.

`vDSP_fft2d_zipD`

Computes a 2D forward or inverse in-place, double-precision complex FFT.

In-Place FFT Functions with Temporary Buffer

`vDSP_fft2d_zipt`

Computes a 2D forward or inverse in-place, single-precision complex FFT using a temporary buffer.

`vDSP_fft2d_ziptD`

Computes a 2D forward or inverse in-place, double-precision complex FFT using a temporary buffer.

See Also

Functions for 2D Complex FFT

Out-of-Place Functions for 2D Complex FFT

Perform fast Fourier transforms out of place on 2D complex data.