fir

keys:	**[from]**	*<number>*	source picture
	[to]	*<number>*	output picture
	with	*<number>*	picture containing filter kernel
	radius	*<number>*	if ***gaussian***, rms kernel radius
options:	**add/subtract**		produces filter output only, added to or subtracted from source
	modulus		if ***add/subtract***, produce modulus of output
	separable		apply 1-D kernel separately in *x* and *y* directions
	gaussian/laplacean		use gaussian profile smoothing kernel, or 3x3 point laplacean kernel

Use **fir** to apply *fir* filters (small block convolution operators), which replace each pixel by a weighted average of its neighbours. You can define the kernel weights in a small subsidiary picture, or use the standard forms generated internally by **fir**.

Examples

```
fir 1 to 2 with 51
```

This command produces a picture 2 by applying the kernel in picture 51 to picture 1.

```
fir laplacean subtract to display
```

This command displays a sharpened version of the current picture.

```
fir laplacean modulus
```

This command obtains the magnitude of the laplacean of the current picture. This is a rather noise sensitive form of edge operator.

```
fir display gaussian radius 2
```

This command smooths the picture display.

```
create 90 size 5,1; p -2=1,2,3,2,1; fir 1 with 90 separable
```

This command defines a 1-D smoothing kernel and applies it to picture 1 in *x* and *y* directions independently.

fir

Description

If you supply a kernel for the **fir** command, note that kernels need not be square, and may be up to 21 points wide. For large kernels you may well find it faster to effect the convolution by multiplying the *Fourier* transform of the source picture by that of the kernel (embedded in a zero picture of the same size).

There are three ways of defining a kernel:

- use the **with** key and supply your own kernel.
- you can use the **gaussian** option, which generates a square kernel of the form:

$$e^{-\left(\frac{d^2}{2r^2}\right)}$$

  for distance d from the centre, truncated beyond e^{-1} with a minimum size of 3, and normalised to a unit sum (which preserves the source mean). The kernel is in fact separated for speed.

- you can use the **laplacean** option, which invokes a fixed kernel of:

$$\frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & -12 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad \text{or} \quad [\,1 \;\; -2 \;\; 1\,] \quad \text{for 1-D source pictures}$$

If your kernel has the form:

$k(x,y)=p(x)p(y)$

that is, it separates into identical x and y factors, you will find execution is faster if you supply the separated form only and use the **separable** option. The last command given in the **Examples** section, uses a 1-D kernel (1 2 3 2 1) and applies an equivalent kernel:

$$\begin{bmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 4 & 6 & 4 & 2 \\ 3 & 6 & 9 & 3 & 1 \\ 2 & 4 & 6 & 4 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{bmatrix}$$

With each kernel, you can specify the **add** or **subtract** options, which adds or subtracts the filter output to or from the original picture. This is done in most cases simply by adjustment of the supplied kernel without additional computation.

fir

Note that while a kernel of a size up to 21 square is acceptable, the **fir** command processes square, horizontal or vertical 3- and 5- point kernels more efficiently (that is, 5x5, 5x1, 1x5, 3x3, 3x1 and 1x3 kernels). Edge pixels are treated as if the edge pixels of the source continued indefinitely outwards.

Notes

multi-layer pictures: layers processed independently
forms used internally: fp

Defaults and Ranges

keys/options	defaults	range
[from]	current picture, held in the variable *select*	valid picture number
[to]	source picture	valid picture number
with	*none*	valid picture number
radius	radius 1	positive real number
add/subtract	apply the kernel to the source picture	