

# Performance of 3D Depth Migration Algorithms

Jan Thorbecke and Ben Geesink

Thursday June 10 1999;  
Presentations: 9:20, 9:45, 10:10

sgt

## Extrapolation techniques discussed

Slide 1

- 2-dimensional convolution, Blacquièrè et al. (1989)
- Hale-McClellan recursion, Hale (1991)
- series expansion in  $\cos(kr)$  Thorbecke (1997)
- phase screen Stoffa et al. (1990)
- finite-difference Li (1991)

Every method is explained briefly and implementation details of the algorithms are discussed. Numerical examples are given for impulse responses. Finally, we demonstrate how each of the algorithms can benefit from the use of multiple processors.

**Rayleigh II integral:**

$$P^+(\mathbf{x}, \omega) = \int_{\partial D} G^+(\mathbf{x}, \mathbf{x}', \omega) P^+(\mathbf{x}', \omega) d^2 \mathbf{x}'$$

**Phase shift operator**

$$\tilde{G}^+(k_x, k_y, \omega, \Delta z) = \exp(j \sqrt{\frac{\omega^2}{c^2} - (k_x^2 + k_y^2)} \Delta z)$$

where  $P^+(\mathbf{x}', \omega)$  represents the measured wave field at the surface  $\partial D$  and  $G^+(\mathbf{x}, \mathbf{x}', \omega)$  represents the extrapolation operator (Greens function) from the surface  $\partial D$  to a point in the subsurface  $\mathbf{x}$  ?.

In homogeneous media the one-way extrapolation operator in the  $\mathbf{k} - \omega$  (wavenumber-frequency) domain is a simple analytical function which is given by ?:

$$\tilde{G}(k_x, k_y, \omega, \Delta z) = \exp(j \sqrt{\frac{\omega^2}{c^2} - (k_x^2 + k_y^2)} \Delta z) \quad (1)$$

with  $\Delta z$  being a small extrapolation step and  $c$  the propagation velocity of the medium. The advantage of using the phase shift operator in the  $k_x, k_y - \omega$  domain is that the extrapolation result is obtained by multiplication of the data with the phase shift operator. However, multiplication in the  $k_x, k_y - \omega$  domain rules out the possibility of applying a laterally varying operator. To allow laterally varying medium functions a convolution operator in the  $x, y - \omega$  (space-frequency) domain should be used. This spatial convolution operator must be designed in such a way that it gives accurate and stable results within a reasonable computational time. To arrive at this goal two steps must be taken; the first step is an optimum *design* of the spatial operator and the second step deals with a fast *implementation* of the spatial convolution. The most efficient algorithms combine these two steps and a spatial operator is designed in such a way that it can be implemented in a fast way. Note that the extrapolation operator is circular symmetric which makes an efficient optimization and implementation possible.

**Approximations to the phase-shift operator**

Slide 3

$$\begin{aligned}
 \tilde{G}(k_x, k_y) &= \exp(jk_z \Delta z) \\
 &\approx \sum_{m=0}^M \sum_{n=0}^N G_{mn} \cos(k_x m \Delta x) \cos(k_y n \Delta y) \\
 &\approx \sum_{l=0}^L G_l T_l(\cos(k_r \Delta x)) \\
 &\approx \sum_{j=0}^J Y_j [\cos(k_r \Delta x)]^j
 \end{aligned}$$

Most spatial extrapolation methods can be expressed in the wavenumber domain as an approximation to the phase shift operator of equation (1). Different approximations can be based on a power series or on an expansion with respect to the cosine terms of the Fourier transform. Transforming these expansions, with a limited number of terms and in an optimal way, to the spatial domain gives the spatial convolution operator. Three types of expansions are discussed:

$$\begin{aligned}
 \tilde{G}(k_x, k_y) &= \exp(jk_z \Delta z) \\
 &\approx \sum_{m=0}^M \sum_{n=0}^N G_{mn} \cos(k_x m \Delta x) \cos(k_y n \Delta y) \tag{2}
 \end{aligned}$$

$$\approx \sum_{l=0}^L G_l T_l(\cos(k_r \Delta x)) \tag{3}$$

$$\approx \sum_{j=0}^J Y_j [\cos(k_r \Delta x)]^j \tag{4}$$

with  $k_r = \sqrt{(k_x^2 + k_y^2)}$  and  $k_z = \sqrt{k^2 - k_r^2}$ . Note that there are many more expansions possible like the expansions in Laplacian  $(k_x^2 + k_y^2)$ ? or  $k_z$  terms.

Equation (2) represents the inverse Fourier transform of a symmetric (in  $x$  and  $y$ ) spatially limited operator.  $G_{mn}$  are the coefficients of the 2-D spatial convolution operator. The coefficients  $G_{mn}$  used in this paper are obtained by a Weighted Least Squares optimization method. The spatial dimensions of the convolution operator are  $(2M - 1)^2$ .

In equation (3) the 2-dimensional problem is reduced to a 1-dimensional filter problem using the circular symmetry of the 2-D phase shift operator Hale (1991). This method is represented by an Chebychev polynomial ( $T_l$ ) in 1-dimensional cosine terms. The coefficients  $G_l$  represent the 1-D phase-shift operator and are obtained with any preferred 1-D

optimization method. The cosine terms  $\cos(k_r \Delta x)$  are approximated by small ( $S \times S$ ) 2-D convolution filters. The spatial size of the operator is  $(L(S-1) - S + 2)^2$ .

Equation (4) is a series expansions in  $\cos(k_r \Delta x)$ . The cosine terms are approximated by small ( $S \times S$ ) 2-D convolution filters. The coefficients  $Y_j$  in the series expansions are obtained by numerically optimizing the coefficients given the approximation of the cosine terms. Note that the number of expansion terms is more limited by the accuracy of the floating point implementation than the Chebychev recursion. The spatial size of the operator is also  $(J(S-1) - S + 2)^2$ .

sgt

### Approximations to the one-way wave equation

$$\frac{\partial P(\mathbf{x}, \omega)}{\partial z} \approx j \sqrt{\frac{\omega}{c(\mathbf{x})} + \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}} P(\mathbf{x}, \omega)$$

Slide 4

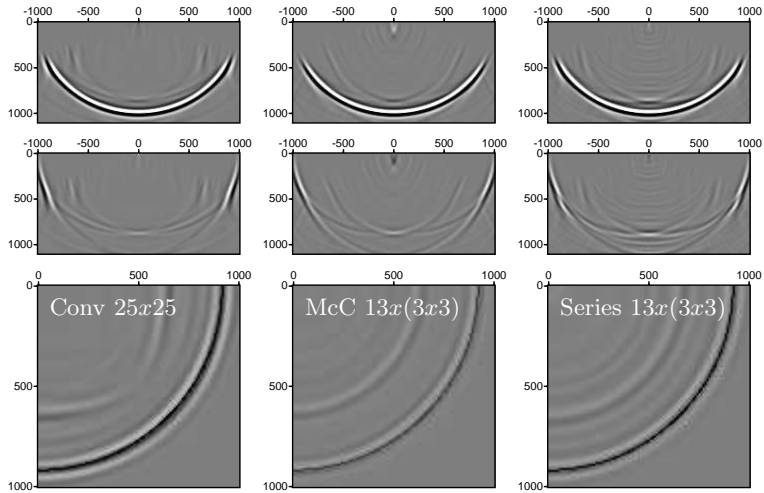
$$\approx j \frac{\omega}{c(\mathbf{x})} \left[ 1 + \sum_{i=1}^n \frac{\alpha_i S_x}{1 + \beta_i S_x} + \sum_{i=1}^n \frac{\alpha_i S_y}{1 + \beta_i S_y} P(\mathbf{x}, \omega) \right]$$

$$S_x = \frac{c^2(\mathbf{x})}{\omega^2} \frac{\partial^2}{\partial x^2}, S_y = \frac{c^2(\mathbf{x})}{\omega^2} \frac{\partial^2}{\partial y^2}$$

$$P(\mathbf{x}, z + dz, \omega) \approx \exp(jk(\frac{\bar{c}}{c(\mathbf{x})} - 1)\Delta z) FT_{\mathbf{k}} \left[ \exp(jk_z \Delta z) \tilde{P}(\mathbf{k}, z, \omega) \right]$$

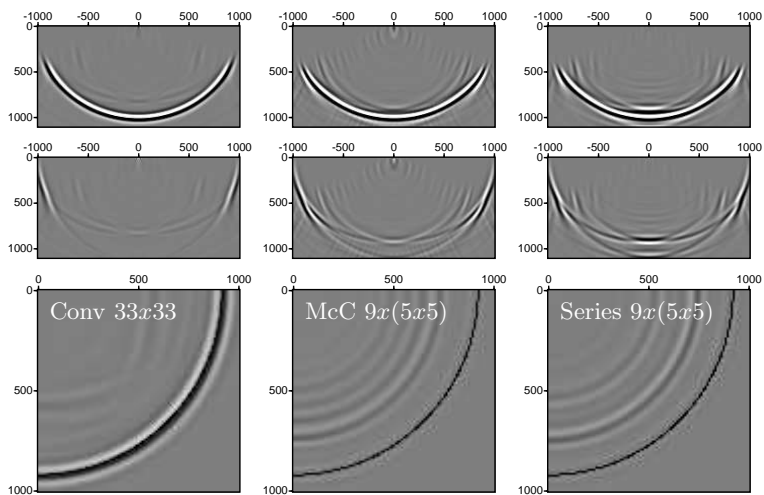
sgi

### Impulse response (25x25)



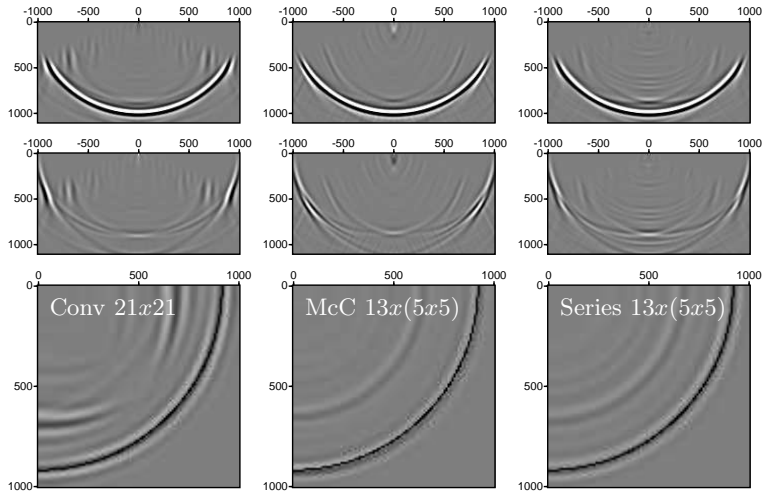
sgi

### Impulse response (33x33)



sgi

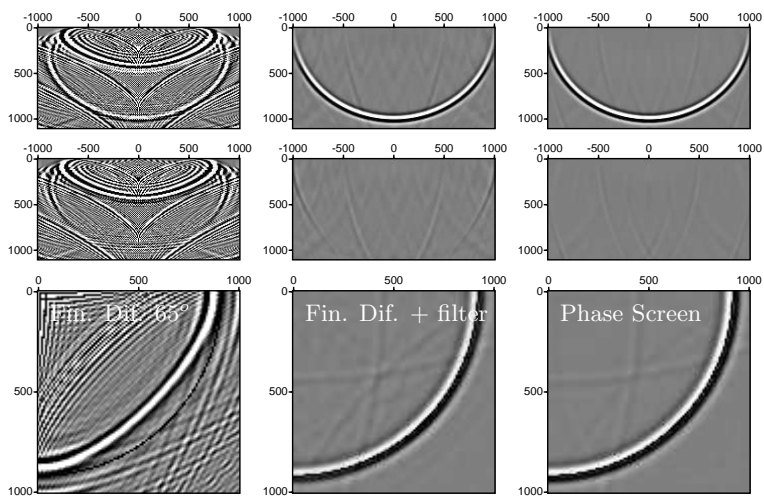
Impulse response ( $21 \times 21 + 13 \times (5 \times 5)$ )



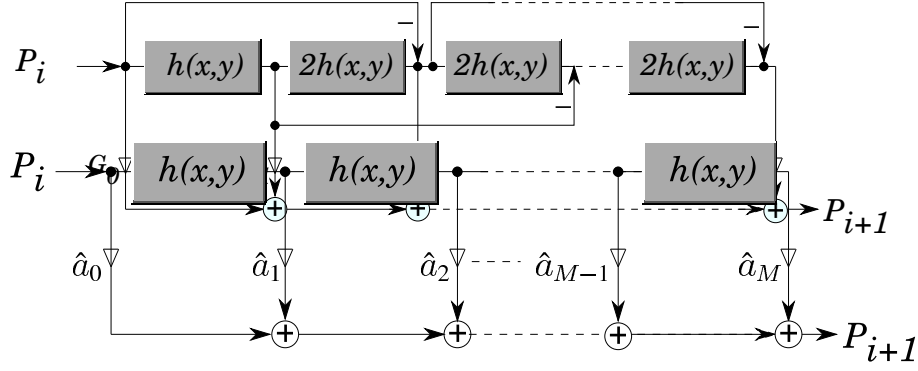
Slide 7

sgi

Impulse response



Slide 8



To check the accuracy of the extrapolation method an impulse response is calculated with  $\Delta x = \Delta y = \Delta z = 10$  m,  $c = 1000$  m/s and a Ricker wavelet with a time delay of 1.0 seconds and a frequency peak at 10 Hz. The impulse response shows us three things; the circularity of the operator, the numerical artifacts and the result at high angles.

To have a fair comparison between the different methods the footprint of the spatial operator is chosen to be the same. For the 2D convolution we have used a  $33 \times 33$  convolution operator and for the Chebychev recursion and series expansion we used a  $3 \times 3$  operator for the cosine operator and 17 terms in the expansion. In Figure ?? the impulse responses are shown for the three spatial convolution methods. Note that all methods give comparable results; the depth slice at  $67^\circ$  of the 2D convolution shows a higher (more accurate) amplitude than the other methods. Note that a higher accuracy for the cosine methods can be achieved by optimizing the cosine terms for every wavenumber  $k = \frac{\omega}{c}$ . However, the performance will suffer from the fact that the  $\cos(kr)$  stencil depends on the local velocity. Using a larger stencil ( $5 \times 5$ ) and less terms (9,) to have the same footprint, gives non-accurate results. A  $13 \times (5 \times 5)$  operator gives comparable results but has a much larger footprint.

In recursive depth extrapolation the (complex) computations are carried out from the  $xy$  plane at depth  $z$  to the  $xy$  plane at depth  $z + \Delta z$ . The secondary cache of the MIPS R10k, which is used in the Origin 2000, has a size of 4 MB and can contain one  $xy$  plane (of complex numbers) of  $724 \times 724$  samples or 2 planes of  $512 \times 512$ . Larger or more depth planes will cause cache misses and introduces a higher latency on load/store operations. The R10K has 32 64-bit floating point registers and can do one multiply-add and one load/store operations at every clock-cycle. Taking these hardware constraints into account the implementation of the different extrapolation methods was carried out:

- The implementation of the 2D spatial convolution operators uses the symmetry in  $x$  and  $y$  explicitly. To reduce misses from the cache the data is rearranged in such a way that the convolution can be done within the cache.

```
for (iy = 0; iy < ny; iy++) {
  for (ix = 0; ix < nx; ix++) {
    for (j = 0; j < opersize; j++) {
      data[iy*nx+ix] += (tmp3[index3+j] + tmp4[index4+j])*hopx[j];
    } } }

```

- The Hale McClellan method is implemented as a Chebychev recursion scheme as shown in Figure ?. The recursion scheme is not very cache friendly because three copies of the  $xy$  planes are needed to calculate the plane at  $z + \Delta z$ . The basic scheme is given by:

```
for (o = 0; o < order; o++)
  for (iy = 0; iy < ny; iy++)
    for (ix = 0; ix < nx; ix++)
      term3[iy*nx+ix] = 2.0*term2[iy*nx+ix] - term1[iy*nx+ix];
      data[iy*nx+ix] += op[o]*term3[iy*nx+ix];
} } }

```

Note that all arrays in these calculations are complex. The term[1,2,3] arrays contain the pre-computed results of the small 2D convolutions of  $\cos(k_r \Delta x)$  at different orders.

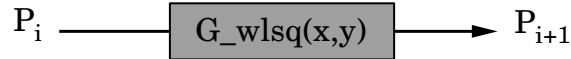
- The series expansion in  $\cos(k_r \Delta x)$ , also shown in Figure ??, is a straightforward implementation of the small 2D convolutions without the extra storage as needed in the Chebychev recursion scheme.

```
for (o = 0; o < order; o++) {
  for (iy = 0; iy < nyo; iy++) {
    for (ix = 0; ix < nxo; ix++) {
      data[iy*nx+ix] += a_m[o]*term1[iy*nx+ix];
    } } }

```

sgi

### 2D-Convolution



Slide 9

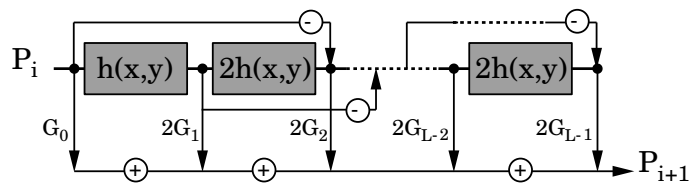
```

for (iy = 0; iy < ny; iy++) {
  for (ix = 0; ix < nx; ix++) {
    for (j = 0; j < opersize; j++) {
      data[iy*nx+ix] += (tmp3[index3+j] + tmp4[index4+j])*hopx[j];
    } } }

```

sgi

### Hale McClellan



Slide 10

```

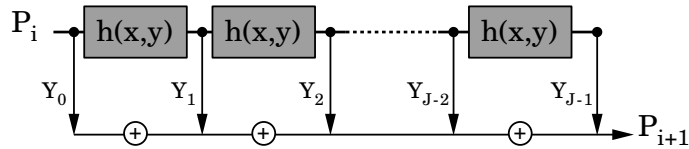
for (o = 0; o < order; o++) {
  for (iy = 0; iy < ny; iy++) {
    for (ix = 0; ix < nx; ix++) {
      term3[iy*nx+ix] = 2.0*term2[iy*nx+ix] - term1[iy*nx+ix];
      data[iy*nx+ix] += op[o]*term3[iy*nx+ix];
    } } }

```



sgt

### Series



Slide 11

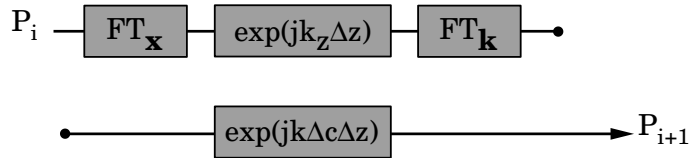
```

for (o = 0; o < order; o++) {
  for (iy = 0; iy < nyo; iy++) {
    for (ix = 0; ix < nxo; ix++) {
      data[iy*nx+ix] += a_m[o]*term1[iy*nx+ix];
    } } }

```

sgt

### Phase-Screen

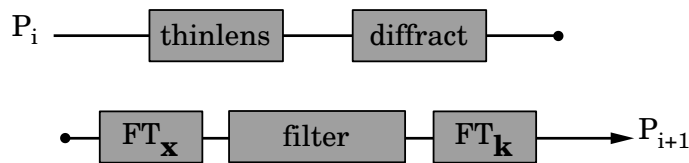


Slide 12

$$\Delta c = 1/c(x,y,z) - 1/c(z)$$

**Finite Difference**

Slide 13



The table above shows the kind and number of operations for every  $(x, y)$  point in the  $xy$  plane of the most inner-loop in the convolution. From this table it can be concluded that the 2D convolution is bound by the number of load/store operations. Note that it is possible to implement the 2D convolution in such a way (by not using the symmetry in the operator) that it will be floating point bound. Although this code will run closer to maximum performance, it will also use more operations and it will take more time for the same task. The Hale McClellan scheme is also bound by the number of load/store operations and can only be run at 50% of the floating point peak performance. The series expansion is the only scheme which is bound by the number of floating point operations.

In the migration the extrapolation takes 95 % of the total migration time. The other 5% is taken by the computation of the operator table and IO. In the table below the number of flops are shown for the impulse response calculation for the different methods:

sgi

**Results small model on R10K 195 MHz**

model size: 201 \* 201 \* 111( $nx * ny * nz$ )

Method	Size	table (s)	migr (s)	misc (s)	tot (s)	Mf/s
Conv	25x25	286.3	2104.5	0.7	2391.5	-
McC	13x(3x3)	0.1	946.2	0.7	946.9	-
Series	13x(3x3)	44.6	802.5	0.7	847.8	-
Conv	33x33	625.0	3480.7	0.7	4106.4	-
McC	9x(5x5)	0.1	906.1	0.7	906.9	-
Series	9x(5x5)	30.6	812.1	0.6	843.3	-
PSPC	-	0.0	532.3	0.7	533.0	-
FinDif	-	0.0	672.3	0.7	673.0	-

Slide 14

sgi

**Results medium model on R10K 195 MHz**

model size: 2001 \* 1201 \* 7( $nx * ny * nz$ )

Method	Size	table (s)	migr (s)	misc (s)	tot (s)	Mf/s
Conv	25x25	293.3	7786.9	13.1	8093.3	-
McC	13x(3x3)	0.1	5172.4	12.5	5185.0	-
Series	13x(3x3)	218.4	4313.9	12.9	4545.2	-
Conv	33x33	640.7	11971.	12.8	12625.	-
McC	9x(5x5)	0.1	4454.6	12.6	4467.3	-
Series	9x(5x5)	142.9	3852.7	12.6	4008.2	-
PSPC	-	0.0	2324.1	13.0	2337.1	-
FinDif	-	0.0	3594.5	14.8	3609.3	-

Slide 15

sgi

**Compare single CPU: R10K 195, 250 + R12K 300 MHz**

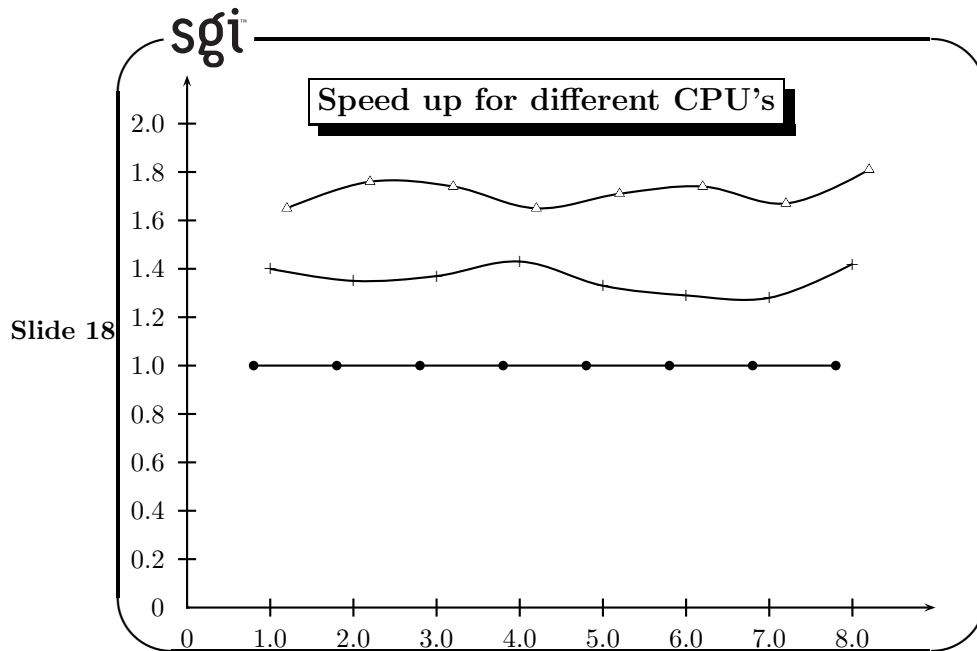
Slide 16

model size:  $2001 * 1201 * 7(nx * ny * nz)$

sgi

Slide 17

		small			medium		
Method	Size	195	250	300 (vs250)	195	250	300 (vs250)
Conv	25x25	1.0	1.40	1.65 (1.18)	1.0	1.27	1.65 (1.29)
McC	13x(3x3)	1.0	1.35	1.76 (1.30)	1.0	1.54	1.26 (0.82)
Series	13x(3x3)	1.0	1.37	1.74 (1.26)	1.0	1.52	1.31 (0.86)
Conv	33x33	1.0	1.43	1.65 (1.15)	1.0	1.25	1.64 (1.31)
McC	9x(5x5)	1.0	1.33	1.71 (1.28)	1.0	1.29	1.33 (1.03)
Series	9x(5x5)	1.0	1.29	1.74 (1.34)	1.0	1.44	1.43 (1.00)
PSPC	-	1.0	1.28	1.67 (1.30)	1.0	1.23	1.47 (1.19)
FinDif	-	1.0	1.42	1.81 (1.27)	1.0	1.31	1.50 (1.14)



Slide 19

sgi

**Results small model on R10K 250 MHz**

model size: 201 \* 201 \* 111( $nx * ny * nz$ )

Method	Size	table (s)	migr (s)	misc (s)	tot (s)	Mf/s
Conv	25x25	115.5	1596.9	0.7	1713.1	290
McC	13x(3x3)	0.1	699.9	0.6	700.7	165
Series	13x(3x3)	33.2	581.7	0.7	615.6	200
Conv	33x33	253.8	2612.3	2.2	2868.3	305
McC	9x(5x5)	0.1	680.2	0.7	681.0	180
Series	9x(5x5)	22.9	630.9	0.6	654.4	190
PSPC	-	0.0	414.8	0.6	415.4	175
FinDif	-	0.0	471.1	0.6	471.7	200

nxv=201 dxv=10 nyv=201 dyv=10 nzv=11 dzv=10 fmin=5 fmax=5 dstep=10

		cycles	flops	seconds
Conv	25x25	1389070136	1659536661	2.293
McC	13x(3x3)	257262027	181424747	1.040
Series	13x(3x3)	489700519	371343254	0.818
Conv	33x33	2647420950	3466367284	3.733
McC	9x(5x5)	241696370	192538144	1.016
Series	9x(5x5)	413381696	313259119	0.918
Phase screen		155578346	108577671	0.595
Finite Difference		159197841	148239311	0.715

nxv=201 dxv=10 nyv=201 dyv=10 nzv=21 dzv=10 fmin=5 fmax=5 dtsep=20

		cycles	flops	seconds
Conv	25x25	2031708960	2358601265	
McC	13x(3x3)	509045729	362559134	
Series	13x(3x3)	700171619	543454491	
Conv	33x33	3698230648	4654437895	
McC	9x(5x5)	477980469	384835411	
Series	9x(5x5)	631730703	494245936	
Phase screen		305982404	217037153	
Finite Difference		313223405	296360433	

nz=10 nw=1

		flops migr	flops table	MF/s
Conv	25x25	699064604	960472057	291
McC	13x(3x3)	181134387	290360	166
Series	13x(3x3)	172111237	199232017	200
Conv	33x33	1188070611	2278296673	303
McC	9x(5x5)	192297267	240877	181
Series	9x(5x5)	180986817	132272302	188
Phase screen		108459482	118189	173
Finite Difference		148121122	118189	197

sgi

**Parallel results small model on R10K 250 MHz**

model size: 201 \* 201 \* 111( $nx * ny * nz$ )

CPU's	table (s)	migr (s)	comm (s)	tot (s)	scaling
1	115.5	1596.9	0.2	1713.1	1.00
2	58.0	801.2	0.5	860.2	1.99
4	29.3	400.9	2.2	433.0	3.96
8	15.6	198.3	2.4	216.8	7.90
16	8.8	101.9	2.3	113.7	15.0
32	5.5	51.4	2.8	60.5	28.3

Slide 20

sgi

**Results medium model on R10K 250 MHz**

model size: 2001 \* 1201 \* 7( $nx * ny * nz$ )

Method	Size	table (s)	migr (s)	misc (s)	tot (s)	Mf/s
Conv	25x25	448.6	5889.4	13.7	6351.7	260
McC	13x(3x3)	0.1	3360.0	14.2	3374.3	110
Series	13x(3x3)	166.4	2789.1	18.6	2974.1	125
Conv	33x33	972.2	9096.2	13.4	10082	285
McC	9x(5x5)	0.1	3447.2	13.6	3460.9	130
Series	9x(5x5)	111.9	2643.3	14.4	2769.6	140
PSPC	-	0.0	1871.4	22.3	1893.7	130
FinDif	-	0.0	2735.2	18.7	2753.9	135

Slide 21

nxv=2001 dxv=10 nyv=1201 dyv=10 nzv=2 dzv=10 fmin=5 fmax=5

		cycles	flops	seconds
Conv	25x25	6474333504	7922247490	
McC	13x(3x3)	1497153863	1065195911	
Series	13x(3x3)	2554199170	1844055181	
Conv	33x33	11966326268	15938236438	
McC	9x(5x5)	1370365146	1137305603	
Series	9x(5x5)	2170610967	1625743155	
Phase screen		784997848	609343983	
Finite Difference		944664815	961634138	

nxv=2001 dxv=10 nyv=1201 dyv=10 nzv=3 dzv=10 fmin=5 fmax=5

		cycles	flops	seconds
Conv	25x25	10257457988	12071459860	
McC	13x(3x3)	2967973902	2125267290	
Series	13x(3x3)	3804178606	2851069970	
Conv	33x33	18150526816	22991455023	
McC	9x(5x5)	2714435379	2269537798	
Series	9x(5x5)	3450707766	2690518819	
Phase screen		1543911914	1213735299	
Finite Difference		1863245629	1918315609	

nz=1 nw=1

		flops migr	seconds	MF/s
Conv	25x25	4149212370	15.291	258
McC	13x(3x3)	1060071379	9.363	108
Series	13x(3x3)	1007014789	7.646	125
Conv	33x33	7053218585	23.705	284
McC	9x(5x5)	1132232195	8.166	132
Series	9x(5x5)	1064775664	7.327	139
Phase screen		604391316	4.416	130
Finite Difference		956681471	6.823	133



sgi

**Parallel results medium model on R10K 250 MHz**

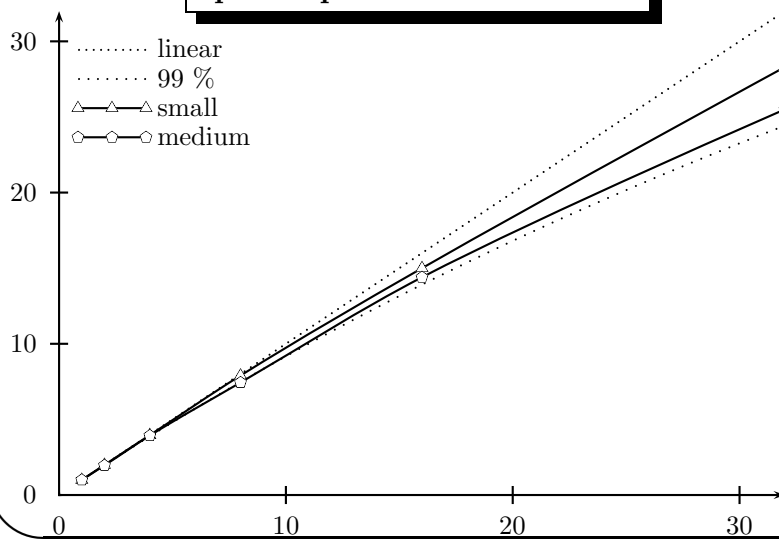
model size:  $2001 * 1201 * 7(nx * ny * nz)$  for Convolution (25x25)

CPU's	table (s)	migr (s)	comm (s)	tot (s)	scaling
1	448.6	5889.4	11.8	6351.7	1.00
2	229.6	2979.6	34.1	3247.0	1.96
4	117.7	1486.4	12.1	1618.1	3.93
8	63.9	749.4	37.7	853.9	7.44
16	34.3	378.3	26.2	440.9	14.4
32	19.6	196.1	31.3	249.2	25.5

Slide 22

sgi

**Speed up for R10K250 MHz**



Slide 23

sgi

**Results small model on R12K 300 MHz**

model size: 201 \* 201 \* 111( $nx * ny * nz$ )

Method	Size	table (s)	migr (s)	misc (s)	tot (s)	Mf/s
Conv	25x25	173.3	1270.8	0.8	1444.9	365
McC	13x(3x3)	0.1	537.3	0.7	538.1	260
Series	13x(3x3)	26.5	460.5	0.7	487.7	245
Conv	33x33	394.3	2087.4	0.8	2482.5	385
McC	9x(5x5)	0.1	529.9	0.7	530.7	225
Series	9x(5x5)	18.1	466.0	0.7	484.8	251
PSPC	-	0.0	317.7	0.7	318.4	200
FinDif	-	0.0	370.0	1.0	371.0	245

Slide 24

nz=10 nw=1 flops migr time Conv 25x25 699064604 1.816 McC 13x(3x3) 181134387 0.668  
 Series 13x(3x3) 172111237 0.675 Conv 33x33 1188070611 2.943 McC 9x(5x5) 192297267  
 0.816 Series 9x(5x5) 180986817 0.687 Phase screen 108459482 0.515 Finite Difference 148121122  
 0.577

sgi

**Parallel results small model on R12K 300 MHz**

model size: 201 \* 201 \* 111( $nx * ny * nz$ )

CPU's	table (s)	migr (s)	comm (s)	tot (s)	scaling
1	173.3	1270.8	0.0	1444.9	1.00
2	87.1	634.7	0.5	723.2	2.00
4	44.5	319.4	1.5	366.4	3.94
8	22.8	162.0	2.7	188.4	7.67
16	12.2	82.6	3.5	99.3	14.6
32	7.9	47.1	5.2	61.7	23.4
64	8.2	23.4	5.8	39.4	36.7

Slide 25

sgi

**Results medium model on R12K 300 MHz**

model size: 2001 \* 1201 \* 7( $nx * ny * nz$ )

Method	Size	table (s)	migr (s)	misc (s)	tot (s)	Mf/s
Conv	25x25	175.4	4718.7	12.6	4905.7	325
McC	13x(3x3)	0.1	4094.4	17.9	4112.4	120
Series	13x(3x3)	140.9	3303.4	13.4	3457.7	140
Conv	33x33	403.0	7261.2	13.8	7678.0	360
McC	9x(5x5)	0.1	3333.8	18.4	3352.3	150
Series	9x(5x5)	84.2	2678.4	15.0	2777.6	165
PSPC	-	0.0	1567.3	15.4	1582.7	150
FinDif	-	0.0	2383.9	13.3	2397.2	155

Slide 26

nz=1 nw=1 flops migr seconds MF/s Conv 25x25 4149212370 12.158 325 McC 13x(3x3)  
 1060071379 8.300 122 Series 13x(3x3) 1007014789 6.951 138 Conv 33x33 7053218585 18.570  
 362 McC 9x(5x5) 1132232195 7.162 151 Series 9x(5x5) 1064775664 6.158 165 Phase screen  
 604391316 3.903 148 Finite Difference 956681471 5.796 157

sgi

**Parallel results medium model on R12K 300 MHz**

model size:  $2001 * 1201 * 7(nx * ny * nz)$  for Convolution (25x25)

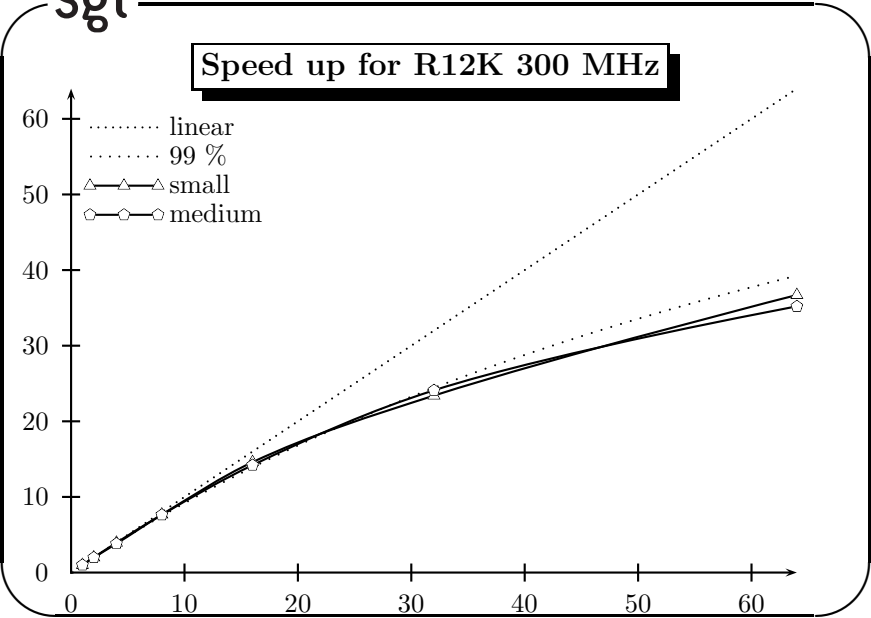
CPU's	table (s)	migr (s)	comm (s)	tot (s)	scaling
1	175.4	4718.7	0.0	4905.7	1.00
2	87.6	2376.2	2.1	2481.8	1.98
4	45.3	1212.6	5.0	1282.2	3.82
8	23.2	602.7	6.9	642.5	7.63
16	11.7	310.6	9.9	345.9	14.2
32	7.0	167.9	13.9	203.8	24.1
64	6.5	101.1	17.3	139.5	35.2

Slide 27

sgi

**Speed up for R12K 300 MHz**

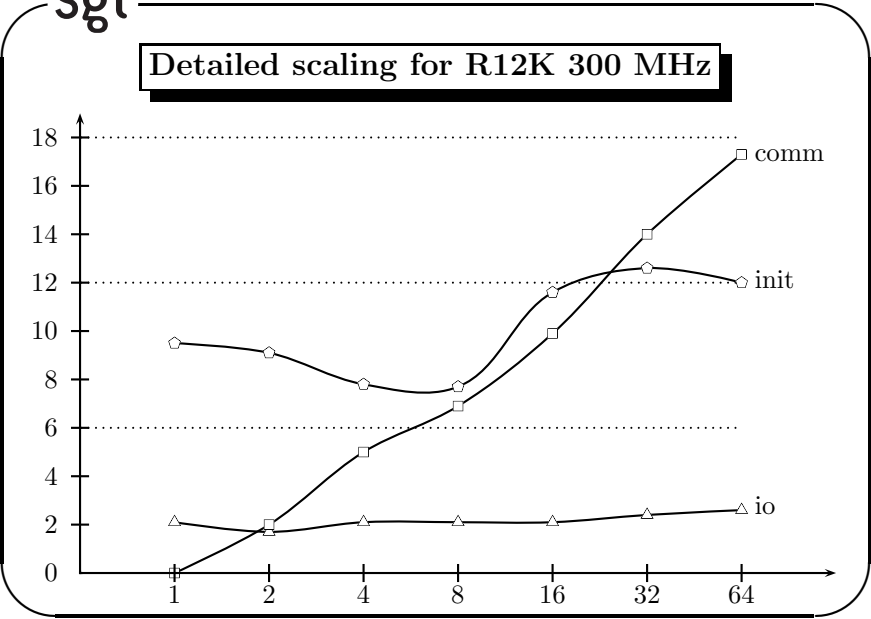
Slide 28



sgi

**Detailed scaling for R12K 300 MHz**

Slide 29



**Comparison**

method	accuracy	circular	table	impl	perf
Conv (33x33)	+	+	-	++	□
Conv (25x25)	+	+	□	++	+
McC 13x(3x3)	□/+	□	+	□	□
McC 9x(5x5)	-	□	++	□	□/+
Series 13x(3x3)	□/+	□/+	□	++	++
Series 9x(5x5)	-	□	+	++	++
PSPC	++	++	++	++	+
FinDif	++	+	++	+	+

Slide 30

Note; accuracy is defined in homogeneous medium.

In the table below a simplified summary is given. The columns in this table have the following meaning:

- *circular*: the circularity of the impulse response.
- *table*: the amount of cycles needed to compute all the operator coefficients needed in the convolution scheme. In the table ++ means a minimum time.
- *implementation*: the simplicity of the implementation.
- *performance*: the performance of the scheme on the Origin 2000.

### Conclusions

Slide 31

- the 2D convolution requires the most multiplications and additions but also makes best use of the hardware
- the performance of the Hale McClellan method suffers from the extra copy needed in the Chebychev recursion scheme
- the series expansion overcomes this problem and can be implemented more efficiently
- PSPC and Finite Difference needs to be tested in inhomogeneous model

### References

Slide 32

- Blacquièrre, G., Debeye, H. W. J., Wapenaar, C. P. A., and Berkhout, A. J., 1989, 3D table-driven migration: *Geophysical Prospecting*, **37**, no. 8, 925–958.
- Hale, D., 1991, 3-D depth migration via McClellan transformations: *Geophysics*, **56**, 1778–1785.
- Li, Z., 1991, Compensating finite-difference errors in 3-D migration and modeling: *Geophysics*, **56**, 1650–1660.
- Stoffa, P. L., Fokkema, J. T., de Luna Freire, R. M., and Kessinger, W. P., 1990, Split-step fourier migration: *Geophysics*, **55**, no. 04, 410–421.
- Thorbecke, J. W., 1997, Common focus point technology: Ph.D. thesis, Delft University of Technology.