NIMBLE comparisons to SMC, POMP, and Biips using fastCopy branch

```
## Derived CmodelBaseClass created by buildModelInterface for model code
```

All of our comparison swere made using Daniel's linear Gaussian model with the correlated parametrization.

fastCopy vs. previous build

We begin by comparing the performance of various particle filters using the fastCopy build of NIMBLE to their performance under the previous devel version. Below are our results for the bootstrap filter. In general, the run-times for both versions of NIMBLE scale linearly with the number of particles. However, the fastCopy build consistently runs between 2 - 3 times faster than the devel branch.

Derived CmodelBaseClass created by buildModelInterface for model code Derived CmodelBaseClass created by buildModelInterface for model code % latex table generated in R 3.2.0 by xtable 1.7-4 package % Tue Jun 23 12:36:02 2015

	Method	N.Particles	Min	Median	Mean	Max
2	NIMBLE devel Bootstrap	100	0	0.015	0.013	0.02
6	NIMBLE fastCopy Bootstrap	100	0	0.015	0.011	0.02
3	NIMBLE devel Bootstrap	1000	0.11	0.115	0.118	0.13
7	NIMBLE fastCopy Bootstrap	1000	0.05	0.07	0.069	0.08
4	NIMBLE devel Bootstrap	10000	1.17	1.215	1.216	1.27
8	NIMBLE fastCopy Bootstrap	10000	0.53	0.6	0.632	0.81
5	NIMBLE devel Bootstrap	50000	6.41	6.93	7.225	9.06
9	NIMBLE fastCopy Bootstrap	50000	2.72	2.835	2.824	2.94

Below are the results for the auxiliary filter. Again, we see the fastCopy build running between 2 - 3 times as fast as the devel build. In addition, for the same number of particles, the auxiliary filter runs about 20% slower than the bootstrap filter.

Derived CmodelBaseClass created by buildModelInterface for model code Derived CmodelBaseClass created by buildModelInterface for model code % latex table generated in R 3.2.0 by xtable 1.7-4 package % Tue Jun 23 12:41:19 2015

	Method	N.Particles	Min	Median	Mean	Max
2	NIMBLE devel Auxiliary Filter	100	0	0.02	0.015	0.02
6	NIMBLE fastCopy Auxiliary Filter	100	0	0.01	0.01	0.02
3	NIMBLE devel Auxiliary Filter	1000	0.14	0.19	0.191	0.22
7	NIMBLE fastCopy Auxiliary Filter	1000	0.06	0.08	0.08	0.09
4	NIMBLE devel Auxiliary Filter	10000	1.74	1.92	1.889	1.98
8	NIMBLE fastCopy Auxiliary Filter	10000	0.87	0.975	0.974	1.06
5	NIMBLE devel Auxiliary Filter	50000	8.82	9.63	9.734	12.02
9	NIMBLE fastCopy Auxiliary Filter	50000	3.6	3.985	4.218	5.61

Below are the results for the Liu and West filter. For the Liu and West filter, for large numbers of particles, the fastCopy build runs about 30% quicker than the devel build. This is possibly because more of the time in the Liu and West filter is spent doing mathematical operations / generating new parameter values than in the other filters.

Derived CmodelBaseClass created by buildModelInterface for model code Warning: sum of weights = 0 in rankSample. Returning a uniform sample. Warning: sum of weights = 0 in rankSample. Returning a uniform sample. Warning: sum of weights = 0 in rankSample. Returning a uniform sample. Derived CmodelBaseClass created by buildModelInterface for model code Warning: sum of weights = 0 in rankSample. Returning a uniform sample. % latex table generated in R 3.2.0 by xtable 1.7-4 package % Tue Jun 23 12:45:33 2015

	Method	N.Particles	Min	Median	Mean	Max
2	NIMBLE devel LW Filter	100	0.06	0.06	0.066	0.08
5	NIMBLE fastCopy LW Filter	100	0	0.055	0.04	0.06
3	NIMBLE devel LW Filter	1000	0.63	0.64	0.644	0.67
6	NIMBLE fastCopy LW Filter	1000	0.54	0.58	0.575	0.61
4	NIMBLE devel LW Filter	10000	6.55	6.595	6.602	6.7
7	NIMBLE fastCopy LW Filter	10000	5.58	5.72	5.712	5.79

In the rest of the document, we rerun our prevous comparisons of NIMBLE to the SMC package, the POMP package, and Biips, using the fastCopy branch,

NIMBLE vs. SMC

For large numbers of particles, NIMBLE now runs between approximately 10 times (for 10,000 particles) and 35 times (for 50,000 particles) faster than the SMC package.

% latex table generated in R 3.2.0 by xtable 1.7-4 package % Tue Jun 23 00:23:04 2015

	Method	N. Particles	Min.	Median	Mean	Max.
1	NIMBLE Auxiliary Filter	100	0	0.01	0.029	0.22
2	SMC R Package Auxiliary Filter	100	0.01	0.02	0.021	0.03
3	NIMBLE Auxiliary Filter	1000	0.04	0.06	0.059	0.07
4	SMC R Package Auxiliary Filter	1000	0.11	0.11	0.114	0.13
5	NIMBLE Auxiliary Filter	10000	0.63	0.66	0.656	0.69
6	SMC R Package Auxiliary Filter	10000	5.12	5.225	5.45	7.42
7	NIMBLE Auxiliary Filter	50000	3.31	3.355	3.359	3.4
8	SMC R Package Auxiliary Filter	50000	119	120.4	121.2	127

NIMBLE vs. POMP

POMP has the Bootstrap filter and Liu and West filters implemented, so we compare them to the NIMBLE implementation.

First, we compare the Bootstrap filter between NIMBLE and POMP. Whereas previously POMP ran significantly faster than NIMBLE for the bootstrap filter, we now see very comparable times across a range of particle values.

% latex table generated in R 3.2.0 by xtable 1.7-4 package % Tue Jun 23 07:31:08 2015

	Method	N. Particles	Mean Log Lik.	Min.	Median	Mean	Max.
1	NIMBLE Bootstrap Filter	100	11.12	0	0.01	0.011	0.05
2	POMP R Package Bootstrap Filter	100	13.46	0.02	0.05	0.05	0.08
3	NIMBLE Bootstrap Filter	1000	18.08	0.07	0.1	0.105	0.17
4	POMP R Package Bootstrap Filter	1000	17.25	0.05	0.085	0.103	0.22
5	NIMBLE Bootstrap Filter	10000	18.55	0.52	0.845	0.875	1.48
6	POMP R Package Bootstrap Filter	10000	18.36	0.4	0.745	0.743	1.03
7	NIMBLE Bootstrap Filter	50000	18.65	2.88	3.27	3.505	4.87
8	POMP R Package Bootstrap Filter	50000	18.68	2.87	3.275	3.579	5.19

Next, we compare the Liu and West filter between NIMBLE and POMP. NIMBLE previously had a speed advantage over POMP for the Liu and West filter, and that advantage has grown with the fastCopy branch.

% latex table generated in R 3.2.0 by xtable 1.7-4 package % Tue Jun 23 10:07:22 2015

	Method	N. Particles	Min.	Median	Mean	Max.
1	NIMBLE Liu and West Filter	10000	5.72	6.445	6.393	7.27
2	POMP R Package Liu and West Filter	10000	15.41	19.54	19	23.09
3	NIMBLE Liu and West Filter	50000	27.67	28.59	29.15	34.92
4	POMP R Package Liu and West Filter	50000	93.17	96.11	101.3	141.3

NIMBLE vs. Biips

Finally, we provide a comparison of NIMBLE to Biips. NIMBLE runs between 5 and 8 times faster than Biips for varying numbers of particles.

% latex table generated in R 3.2.0 by xtable 1.7-4 package % Tue Jun 23 12:45:34 2015

	Method	N. Particles	Min.	Median	Mean	Max.
1	NIMBLE Bootstrap Filter	100	0.01	0.02	0.018	0.02
2	Biips Bootstrap Filter	100	0.09	0.11	0.186	0.9
3	NIMBLE Bootstrap Filter	1000	0.1	0.13	0.123	0.13
4	Biips Bootstrap Filter	1000	0.86	0.89	0.941	1.39
5	NIMBLE Bootstrap Filter	10000	1.3	1.43	1.429	1.5
6	Biips Bootstrap Filter	10000	10.76	10.92	10.97	11.19
7	NIMBLE Bootstrap Filter	50000	6.17	6.32	6.53	7.43
8	Biips Bootstrap Filter	50000	43.85	44.16	46.72	54.36