Load Balancing and Scalability of a Subgrid Orography Scheme in a Global Climate Model

Steven Ghan and Timothy Shippert
Pacific Northwest National Laboratory

revision

International Journal of High Performance Computing Applications

February, 2005

Running Head: Load Balancing Subgrid Orography

Steven J. Ghan (corresponding author)
Climate Physics Group
Mail Stop K9-30
Pacific Northwest National Laboratory
Richland WA 99352
Phone 509-372-6169
Fax 509-372-6168
Email steve.ghan@pnl.gov

Timothy Shippert
Climate Physics Group
Mail Stop K9-30
Pacific Northwest National Laboratory
Richland WA 99352
Phone 509-375-5997
Fax 509-372-6168
Email tim.shippert@pnl.gov

Summary

A subgrid orography scheme has been applied to the National Center for Atmospheric Research Community Atmosphere Model. The scheme applies all of the model column physics to each of up to eleven elevation classes within each grid cell. The distribution of the number of elevation classes in each grid cell is highly inhomogeneous. This could produce a serious load imbalance if the domain decomposition distributes grid cells evenly across processors. But since the distribution of classes is static, static load balancing can be used to distribute the elevation classes uniformly across processors. The load balancing is accomplished first by distributing the number of classes evenly within each process. The number of chunks on processes is distributed uniformly across processes and the dynamics-physics transpose cost is minimized by assigning chunks to processes with the most dynamics grid cells from that chunk. Parallel efficiency with the subgrid scheme and load balancing exceeds parallel efficiency without the subgrid scheme for up to 128 processors. The load balancing across processes decreases runtime by 10–30% depending on configuration.

1. Introduction

A subgrid orography scheme (Ghan et al., 2002) has been applied to the National Center for Atmospheric Research (NCAR) Community Atmosphere Model (CAM3) and Common Land Model (CLM3). CAM3 (Collins et al., 2004) is a global atmospheric circulation code designed to run on a variety of computational platforms, including single processor workstations, shared memory machines, distributed memory systems, symmetric multiple processor (SMP) systems, and most recently on distributed vector systems. The code marches forward in time, alternating between what is called dynamics and physics. The dynamics section of the code solves the conservation equations for mass, momentum, energy, and several forms of water in the atmosphere. Three different numerical integration schemes (dynamic cores, or dycores) for the dynamics are available: Eulerian-Spectral (Simmons and Strüfing, 1983), hereafter EUL, Semi-Lagrange-Spectral (Williamson and Olson, 1994), and Semi-Lagrange-Finite-Volume (Lin and Rood, 1996, 1997), hereafter FV. The physics code calculates the momentum, energy and water source functions needed by the dynamics, vertical column by column, including the fluxes from the surface. The CLM3 (Dai et al., 2003; Oleson et al., 2004) can be considered part of this physics. The physics is normally calculated on the same grid used by the dynamics.

Figure 1 illustrates how the subgrid scheme works. It consists of seven steps. First, a preprocessor uses a high-resolution (typically 1 km) global surface elevation dataset to determine the fractional area and mean elevation for each of a modest set of elevation classes within each model grid cell. Second, this information is used during the global climate simulation to determine the vertical displacement of air parcels passing through grid cells, for each elevation class. Third, conservation of energy and moisture is used to determine the temperature and humidity profiles for each elevation class. Fourth, the full atmospheric and land surface physics of the climate model is applied to these profiles for each elevation class. The atmosphere model and land model use the same grid and elevation classification, so consistency is ensured. Fifth, the area-weighted grid cell mean of the heating and moistening rates are applied to the grid cell mean conservation equations for energy and moisture. Sixth, the full model history is written to disk for each elevation class. Seventh, in postprocessing the model history is distributed according to a high-resolution (typically 5 km) distribution of surface elevation, interpolating between elevation classes to the elevation at high resolution, and between global model grid cells. Further description of the scheme can be found in Ghan et al. (2002).

Figure 2 illustrates the performance of the subgrid scheme. The simulated spatial distribution of annual mean precipitation for the western United States is compared with a gridded distribution of observations (Daly et al., 1994, 1997). The precipitation is simulated with and without the subgrid scheme using the EUL dycore at T42 spectral resolution (about 2.8˚ latitude and longitude) and the FV dycore at 2x2.5 resolution. Also shown is the precipitation simulated without the subgrid scheme using the EUL dycore at T239 resolution (Phillip Duffy and Bala Govindasamy, personal communication). The simulations with the subgrid scheme are clearly superior to the simulations at the same resolution but without the subgrid scheme, and are competitive with the much more expensive simulation at T239 resolution (0.5˚ latitude and longitude).

The elevation range for each class is the same for all grid cells, so the heterogeneous distribution of surface elevation leads to a highly inhomogeneous distribution of the number of elevation classes within each grid cell. Figure 3 illustrates the geographical distribution of the number of elevation classes for one model configuration with a maximum of eleven classes defined by the boundaries at surface elevations of 200, 400, 700, 1000, 1500, 2000, 3000, 4000, 5000, 7000, and 9000 m. Most grid cells have only one class, but some have as many as ten. The entire model column physics (land and atmosphere) is applied to each elevation class, so the column physics computations varies widely from grid cell to grid cell.

This heterogeneous distribution of elevation classes could produce a serious imbalance in the computational burden of the column physics if the domain decomposition distributes grid cells evenly across processors. Model column physics typically comprises 75–80% of the total model run time, so a large load imbalance in the column physics could seriously degrade model performance in a massively parallel simulation. But since the distribution of classes is static, static load balancing can be used to distribute the elevation classes uniformly across processors. In section 2 we describe a load balancing algorithm that we have applied to the column physics in CAM3 and CLM3. In section 3 we evaluate the computational performance of the algorithm in a variety of model configurations. Extrapolation of the performance to higher resolution is addressed in section 4. We summarize our conclusions in section 5.

2. Load Balancing Algorithm

In CAM3 the vast majority of the computations are comprised of model physics and model dynamics. The model physics involves vertical but not horizontal transport, so the physics is always performed column by column, with no communication between columns. The model dynamics involves both horizontal and vertical transport, so communication between MPI processes is required for any domain decomposition.

When the subgrid scheme is used, load balancing is needed both between processes for distributed computing systems and within processes for shared memory and SMP systems. Load balancing within processes does not require any additional interprocess communication, but load balancing between processes does. Load balancing within processes is always treated when the subgrid scheme is used, but interprocess load balancing is treated as an option. In section 3 we will compare timings with and without interprocess load balancing.

To understand the load balancing algorithm it is helpful to understand how the column physics is organized in CAM3. The model physics for a set of columns is assigned to each of a number of “chunks” (Worley and Drake, 2005). The assignment of columns to chunks and of chunks to processes both influence the degree of load balancing across processors. In CAM3, OPENMP is used to assign the same number of chunks to each processor on a given process. For an SMP computing system, the same number of processors is assigned to each of multiple processes. Thus, the load will be balanced if each chunk has the same number of physics columns and if each process has the same number of chunks.

When the subgrid scheme is used, only domain decomposition of physics columns by latitude is permitted. Although the dynamics can still be decomposed by latitude and

level when the finite-volume dycore is used, there is no reason to decompose by level because the physics decomposition limits the number of processes to the number of latitudes. Although this limits the number of processes permitted, it has the benefit of, if interprocess load balancing is not used, permitting the same domain decomposition for physics and dynamics, thus eliminating communication between processes when physics and dynamics interact,.

Load balancing within processes can be achieved by distributing the number of columns assigned to a chunk evenly across all chunks. Assuming this has been done, load balancing between processes can be achieved by distributing the chunks evenly across processes.

In balancing the load within processes, the distribution of the number of elevation classes evenly across all chunks must also ensure that all classes in a grid cell are retained within the same chunk of columns (to permit forming grid cell means of fields using values in each elevation class). This is accomplished by the following algorithm:

1. Rank grid cells according to number of classes.
2. Starting with cells with the most classes, assign all classes in the cell to the chunk with the fewest columns.
3. Continue distribution until classes for all cells have been assigned to chunks.

In computer science parlance this algorithm is known as the “knapsack problem”, in which a heterogeneous collection of rocks is distributed one rock at a time, beginning with the biggest rock, which is given to the fastest kid, iterating until all rocks are distributed. To use this algorithm, the chunks must be large enough to hold all of the classes within a grid cell. The maximum number of elevation classes in a grid cell is typically 10–20, which is comparable to or smaller than the optimal number of columns in chunks for most computers, which is 16 on scalar processor machines and much larger on vector machines.

If load balancing across processes is not performed then the columns can be assigned to the same process that the grid cell for the dynamics is assigned to. This eliminates the need for interprocess communication when physics and dynamics interact. In this case columns are assigned to chunks by drawing from only the set of grid cells and columns from the subdomain assigned to the same process.

If, on the other hand, load balancing across processes is performed then the columns need not be drawn from the same process. Columns are assigned to chunks by drawing from columns in the full global domain.

The load across processes can be balanced by assigning the same number of chunks to each process. However, doing so introduces a communication cost because the columns in the chunks aren’t necessarily from grid cells assigned to the same process that the grid cells for the dynamics are assigned to. To minimize the communication cost associated with the dynamics-physics transpose, the following algorithm is used:

1. Calculate the average number of chunks per process from the total number of chunks and processes.
2. For each chunk, calculate the number of columns assigned to each process.
3. Assign the chunk to the available process with the most columns from that chunk.
4. If the number of chunks on a process reaches the average number of chunks, remove the process from the pool of processes available for assignment.

This algorithm is guaranteed to assign the same, or nearly the same, number of chunks to

each process.

More telling is the fraction of columns assigned to the same process as the grid cell for the dynamics. Figure 4 shows the fraction for a variety of configurations using a particular elevation classification scheme. The fraction for two different resolutions is shown: spectral resolution T85 (about 2.8º latitude and longitude) and finite-volume resolution 2º latitude by 2.5º longitude. For the same number of processes, the fraction is nearly independent of the model resolution. The fraction decreases from one to an asymptotic value just below 0.3. The asymptotic value reflects the level of heterogeneity of surface elevation, and hence is fixed for the current Earth topography. As we shall see, the fact that the asymptote is significantly larger than zero improves the scaling of the load balancing algorithm.

3. Performance

To evaluate the computational performance of the load balancing algorithm, we have performed a series of timing tests on Cheetah, an IBM Power p690 system at the Center for Computational Sciences at Oak Ridge National Laboratory. Cheetah has 27 "Regatta" SMP nodes, each node with thirty-two 1.3 GHz Power4 processors and at least 32 GB of memory. All nodes are connected with an IBM Federation switch. Cheetah has provided many of the cycles needed to run CAM3 (coupled with an ocean model) to simulate global warming for a variety of scenarios, so the performance of CAM3 on Cheetah is pertinent.

A variety of timing tests were performed. Each test consisted of a ten-day simulation, which is long enough to ensure that initialization takes a negligible fraction (at most 4% for 128 p) of the simulation time. Two different dynamic cores were tested: EUL and FV. These dycores were each tested at one resolution: triangular truncation T85 (roughly 2.8º latitude and longitude) for EUL, and 2º latitude by 2.5º longitude for FV. Tests with (11 classes) and without (1 class) the subgrid scheme and with and without interprocess load balancing were performed. Note that even without interprocess load balancing, the load within processes is still balanced.

Although the FV dycore has the ability to use two-dimensional as well as one-dimensional domain decomposition, as explained in section 2 two-dimensional domain decomposition cannot increase the number of processes permitted when the subgrid scheme is used. All FV tests therefore use one-dimensional domain decomposition (by latitude), which limits the number of processes to one third the number of latitudes, which at 2ºx2.5º resolution is 30 processes. The EUL dycore supports decomposition by latitude only. The number of processes is therefore limited to the number of latitudes, which at T85 resolution is 128 processes.

Timing tests are designed to test the dependence of the timing on several parameters. These include the number of processes and processors, load balancing, and the partitioning of the processors into MPI processes versus shared memory processors. The performance may also depend on whether the processes are distributed across multiple physical nodes or concentrated on the minimum number of nodes. All timing tests reported here were run on the minimum number of physical nodes, so we do not explore this dependency.

The application of the subgrid scheme to CAM3 does not degrade the parallel

efficiency. Figure 5 shows the parallel efficiency (the speedup divided by processor count) as a function of processors for one processor per process. A parallel efficiency of one means the speedup is proportional to the number of processors and the run time is inversely proportionate to the number of processors. As expected with the subgrid scheme, the parallel efficiency decreases faster without load balancing than with. Surprisingly, for both dycores the parallel efficiency with the subgrid scheme and load balancing actually exceeds that without the subgrid scheme for all process counts out to 128. This advantage decreases as the process count approaches 128, and is likely to be reversed for higher process counts. But 128 is the maximum number of processes permitted at T85 resolution, so parallel efficiency with the subgrid scheme exceeds that without for all permitted process counts.

Figure 5 shows that the parallel efficiency for the subgrid scheme with load balancing decreases rapidly with process count as the process count approaches 128. This reduction in parallel efficiency is driven by increasing communication cost. Figure 6 shows the cost of physics-dynamics coupling as a function of process count for each configuration. The cost scales very well with or without the subgrid scheme if interprocess load balancing is not used, with the cost as a percent of total run time actually decreasing with process count because no communication is required for the coupling. If interprocess load balancing is used then communication is required for the coupling. The coupling scales well out to about 30 processes, but does not scale as well for more processes. The relative coupling cost nearly doubles between 64 and 128 processes, and the absolute coupling cost actually increases from 13.7 seconds per simulated day for 64 processes to 15.9 for 128 processes. This increased overhead due to communication reduces the benefit of load balancing, but only at the highest number of processes.

Although the physics-dynamics transpose cost is a significant overhead, the load balancing decreases runtime significantly. Figure 7 shows the simulation speed with load balancing relative to the speed without load balancing. Simulations with load balancing run 10–30% faster than simulations without load balancing for all process counts (except 1, of course, for which load balancing is meaningless). Although the benefit varies with process count in unexplained ways, it is clear that as the process count approaches 128 the benefit decreases rapidly.

How does the absolute simulation speed with the subgrid scheme compare with the speed without it? Figure 8 shows the simulation speed with the subgrid scheme with and without load balancing, both normalized by the simulation speed without the subgrid scheme. Simulations without load balancing run about 40–45% of the speed without the subgrid scheme. When load balancing is used, simulations with the subgrid scheme run at about 50% of the speed without the subgrid scheme. Such a speed is consistent with the total number of physics columns, which for both resolutions is about twice with the subgrid scheme as without.

Additional simulations with SMP demonstrate a similar dependence of performance on threads per task with and without the subgrid scheme. On Cheetah optimal performance is typically with 2 threads per task with the subgrid scheme and 4 threads per task without it, but little difference between 2 and 4 in either case.

Worley and Drake (2005) provide timing information for CAM3 on Cheetah without the subgrid scheme. These can be combined with the relative timing in Figure 8 to estimate the absolute timing. We do not report the absolute timing here because it is very

machine-dependent.

4. Performance at Higher Resolution

We have examined the performance of the load balancing algorithm at two resolutions that CAM3 is commonly run at. Future applications are likely to be run at finer resolution. How will finer resolution affect performance? As is common in codes like this, scalability is likely to improve with resolution. In addition, the added computational burden of the subgrid scheme will decrease. Figure 9 shows the number of classes per grid cell averaged over longitude for a variety of resolutions expected for the next decade of simulations with CAM3 and its successors. The same 11-class classification described in section 1 is used for each case. The longitudinal mean represents the computational burden of physics columns for each process if interprocess load balancing is not used. The computational burden of the physics columns dominates the total computations in CAM3, so the number of elevation classes is a reliable predictor of performance if the burden is distributed uniformly across processors. The number of elevation classes decreases monotonically with increasing resolution as more and more elevation variability is explicitly resolved. The importance of load balancing therefore also decreases with increasing resolution.

Averaging over the globe, we can estimate the computational burden of the subgrid scheme from the average number of elevation classes per grid cell. Table 1 shows the average number of classes per grid cell, the grid cell maximum, and the maximum longitudinal mean. The average decreases to 1.33 at 0.25˚x0.3125˚ resolution. In the limit of extremely fine (1 km) resolution, the subgrid scheme disappears. The grid cell maximum number of classes decreases very slowly with resolution, demonstrating the importance of load balancing within as well as across processes. The maximum longitudinal mean is a measure of the importance of load balancing across processes, and also represents the impact of the subgrid scheme on the memory required. It is still more than 2 at 0.25˚x0.3125˚ resolution. Thus load balancing will be important for all resolutions anticipated for the next decade.

5. Conclusions

We have evaluated the computational performance of a load balancing algorithm applied to a subgrid orography scheme operating in a global climate model. The benefit of load balancing within processes could not be evaluated because it is always used with the subgrid orography scheme. Load balancing between processes reduces the run time by 10–30%, with maximum improvement for intermediate process counts. Scalability of the code is significantly improved with load balancing for intermediate process counts, but for more than 100 processes the added communication cost begins to approach the reduction in load imbalance.

Acknowledgements

We are indebted to Pat Worley for his suggestions and for his column distribution code. Phillip Duffy and Bala Govindasamy provided the precipitation simulation at T239

resolution. Two anonymous reviewers provided helpful comments. This work was sponsored by the U.S. Department of Energy Office of Basic Energy Research Scientific Discoveries through Advanced Computing (SciDAC) program. The timing tests were performed at the Oak Ridge National Laboratory Center for Computational Sciences. The Pacific Northwest National Laboratory is operated for the DOE by Battelle Memorial Institute under contract DE-AC06-76RLO 1830.

Biographies

Steven Ghan has worked as a climate modeler at the Pacific Northwest National Laboratory since 1990. Most of his work and interest involve parameterization of cloud-aerosol interactions and subgrid orographic effects on climate. He received his B.S. in Atmospheric Science from the University of Washington in 1979 and his M.S. and PhD. from the Massachusetts Institute of Technology in 1981 and 1988, respectively.

Tim Shippert is a Senior Research Scientist at the Pacific Northwest National Laboratory. His research interests include high performance computing and atmospheric modeling. He has a B.S. in Physics from the Massachusetts Institute of Technology and an M.S. in Physics from the California Institute of Technology.

References

Collins, W. D, Rasch, P. J., Boville, B. A., Hack, J. J., McCaa, J. R., Williamson, D. L., Kiehl, J. T., Briegleb, B., Bitz, C., Lin, S.-J., Zhang, M., and Dai, Y. 2004. Description of the NCAR Community Atmosphere Model (CAM 3.0). NCAR Technical Note NCAR/TN–464+STR.

Dai Y., Zeng, X., Dickinson, R.E., Baker, I., Bonan, G.B., Bosilovich, M.G., Denning, A. S., Dirmeyer, P.A., Houser, P.R., Niu, G., Oleson, K.W., Schlosser, C.A., and Yang, Z.-L. 2003. The Common Land Model. *Bull. Amer. Meteor. Soc*. 84:1013–1023.

Ghan, S. J., Bian, X., Hunt, A. G., and Coleman, A. 2002. The thermodynamic influence of subgrid orography in a global climate model. *Climate Dynamics* 20:31–44, DOI: 10.1007/s00382-002-0257-5.

Lin, S.-J., and Rood, R. B. 1996. Multidimensional flux form semi-lagrangian transport schemes. *Mon. Wea. Rev*. 124:2046–2070.

Lin, S.-J., and Rood, R. B. 1997. An explicit flux-form semi-lagrangian shallow water model on the sphere. *Q. J. R. Meteorol. Soc*. 123:2531–2533.

Oleson, K. W., Dai, Y., Bonan, G., Bosilovich, M., Dickenson, R., Dirmeyer, P., Hoffman, F., Houser, P., Levis, S., Niu, G.-Y., Thornton, P., Vertenstein, M., Yang, Z.-L., and Zeng, X. 2004. Technical Description of the Community Land Model. NCAR Tech. Note NCAR/TN-461+STR.

Simmons, A. J., and Strüfing, R. 1983. Numerical forecasts of stratospheric warming events using a model with hybrid vertical coordinate. *Q. J. R. Meteorol. Soc*. 109:81–111.

Williamson, D. L., and Olson, J. G. 1994. Climate simulations with a semi-Lagrangian version of the NCAR Community Climate Model. *Mon. Wea. Rev*. 122:1594–1610.

Worley, P. H., and Drake, J. B. 2005. Performance portability in the physical parameterizations of the Community Atmospheric Model. *Int. J. High Perf. Comp. Appl*., submitted.

Figure Captions.

Figure 1. Steps in the application of the subgrid scheme. A high-resolution surface elevation dataset (upper left) is used to determine the frequency distribution and mean elevation of a set of elevation classes within each grid cell (upper right). A mountain flow model is used to diagnose the displacement of air parcels with respect to the mean elevation. Conservation of energy and moisture is used to determine the temperature and humidity profiles for each elevation class. Full model physics is applied to those profiles (bottom right), and written to the model history. The history for each elevation class is interpolated according to the high-resolution surface elevation dataset, producing a high-resolution distribution of the climate (bottom left).

Figure 2. The spatial distribution of annual mean precipitation for the western United States as observed at 2.5 minute resolution, as simulated with and without the subgrid scheme using the EUL dycore at T42 spectral resolution and the FV dycore at 2x2.5 resolution, and as simulated without the subgrid scheme using the EUL dycore at T239 resolution.

Figure 3. Number of elevation classes within each 2.8ºx2.8ºgrid cell for an eleven-class classification of surface elevation scheme.

Figure 4. Fraction of physics columns assigned to the same process as the corresponding grid cell for dynamics for T85 spectral resolution and 2ºx2.5º finite-volume resolution.

Figure 5. Parallelization efficiency as a function of processor count for T85 (above) and 2˚x2.5˚ (below) configurations without the subgrid scheme (1 class), with it but without interprocess load balancing (11 class), and with the subgrid scheme and with interprocess load balancing (11 class load balance).

Figure 6. The physics-dynamics coupling time normalized by the total time, as a function of process count for T85 (above) and 2˚x2.5˚ (below) resolution for simulations with and without the subgrid scheme with and without load balancing.

Figure 7. The simulation speed with load balancing normalized by the speed without load balancing as functions of process count for T85 and 2˚x2.5˚ resolution.

Figure 8. The CAM3 simulation speed with the subgrid scheme with and without load balancing, both normalized by the simulation speed without the subgrid scheme, for T85 (above) and 2˚x2.5˚(below) resolution.

Figure 9. The number of elevation classes per grid cell averaged over longitude for a variety of spectral and latitude by longitude resolutions, using the eleven-class classification scheme of Ghan et al. (2002).

Table 1. Global elevation class statistics for a variety of spectral and latitude by longitude resolutions.

	Spectral Resolution				Latitude by Longitude			
	T42	T85	T170	T340	2x2.5	1.25	0.5x0.625	0.25x0.3125
Average classes per cell	2.28	1.85	1.58	1.38	2.10	1.74	1.50	1.33
Maximum classes per cell	11	11	10	9	11	11	10	9
Maximum longitudinal mean classes per cell	3.85	3.16	2.66	2.23	3.64	3.03	2.47	2.06

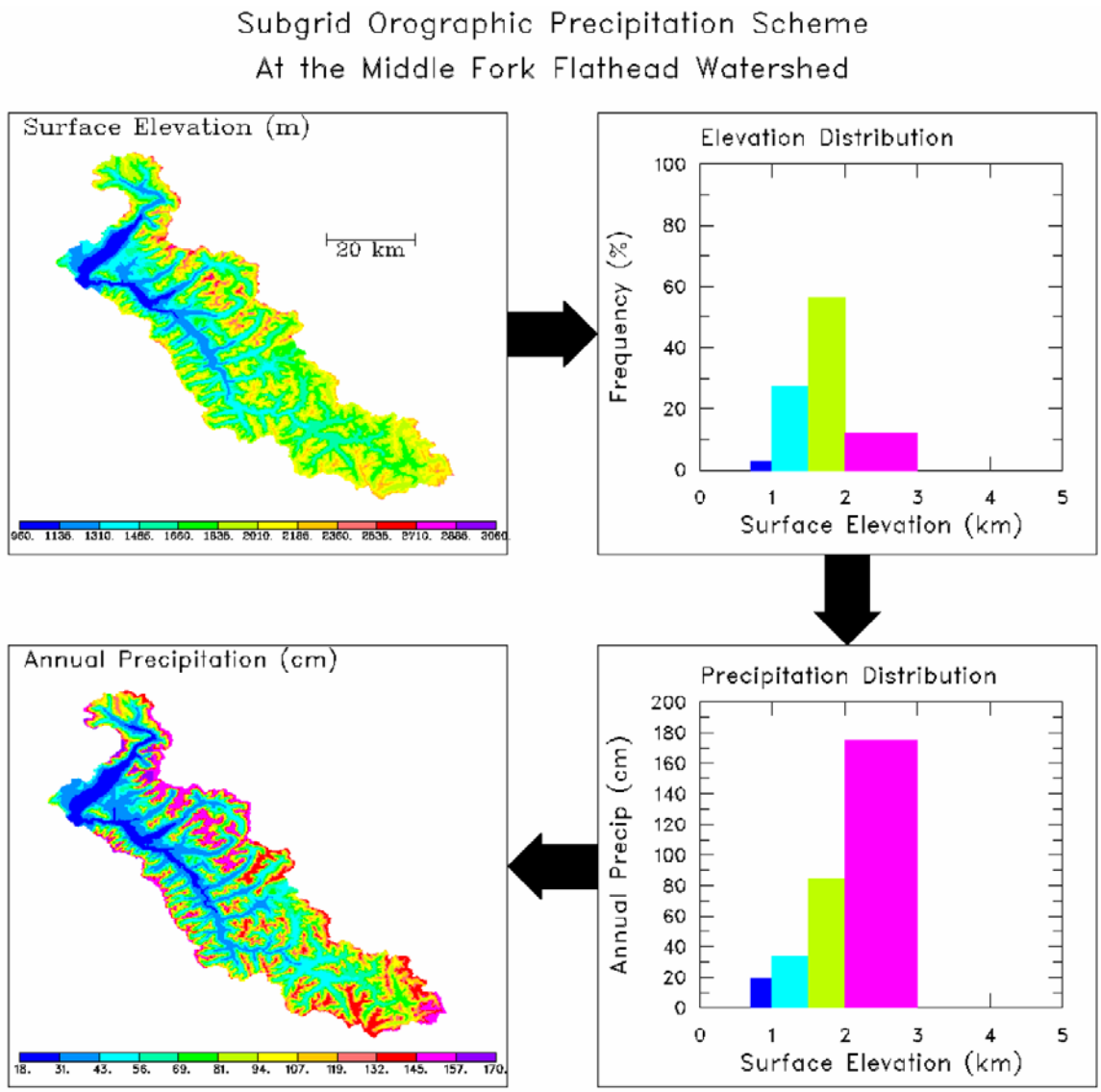


Figure 1. Steps in the application of the subgrid scheme. A high-resolution surface elevation dataset (upper left) is used to determine the frequency distribution and mean elevation of a set of elevation classes within each grid cell (upper right). A mountain flow model is used to diagnose the displacement of air parcels with respect to the mean elevation. Conservation of energy and moisture is used to determine the temperature and humidity profiles for each elevation class. Full model physics is applied to those profiles (bottom right), and written to the model history. The history for each elevation class is interpolated according to the high-resolution surface elevation dataset, producing a high-resolution distribution of the climate (bottom left).

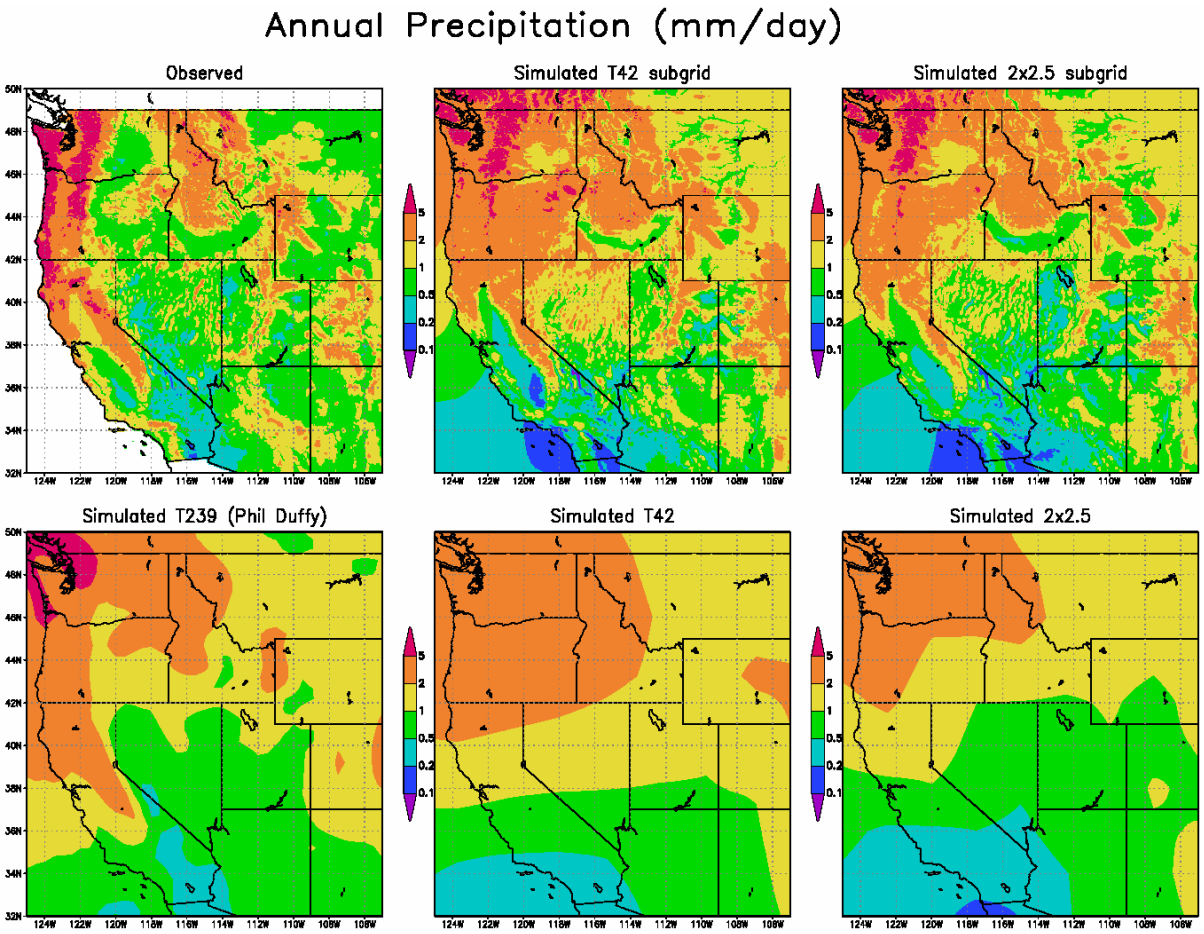


Figure 2. The spatial distribution of annual mean precipitation for the western United States as observed at 2.5 minute resolution, as simulated with and without the subgrid scheme using the EUL dycore at T42 spectral resolution and the FV dycore at 2x2.5 resolution, and as simulated without the subgrid scheme using the EUL dycore at T239 resolution.

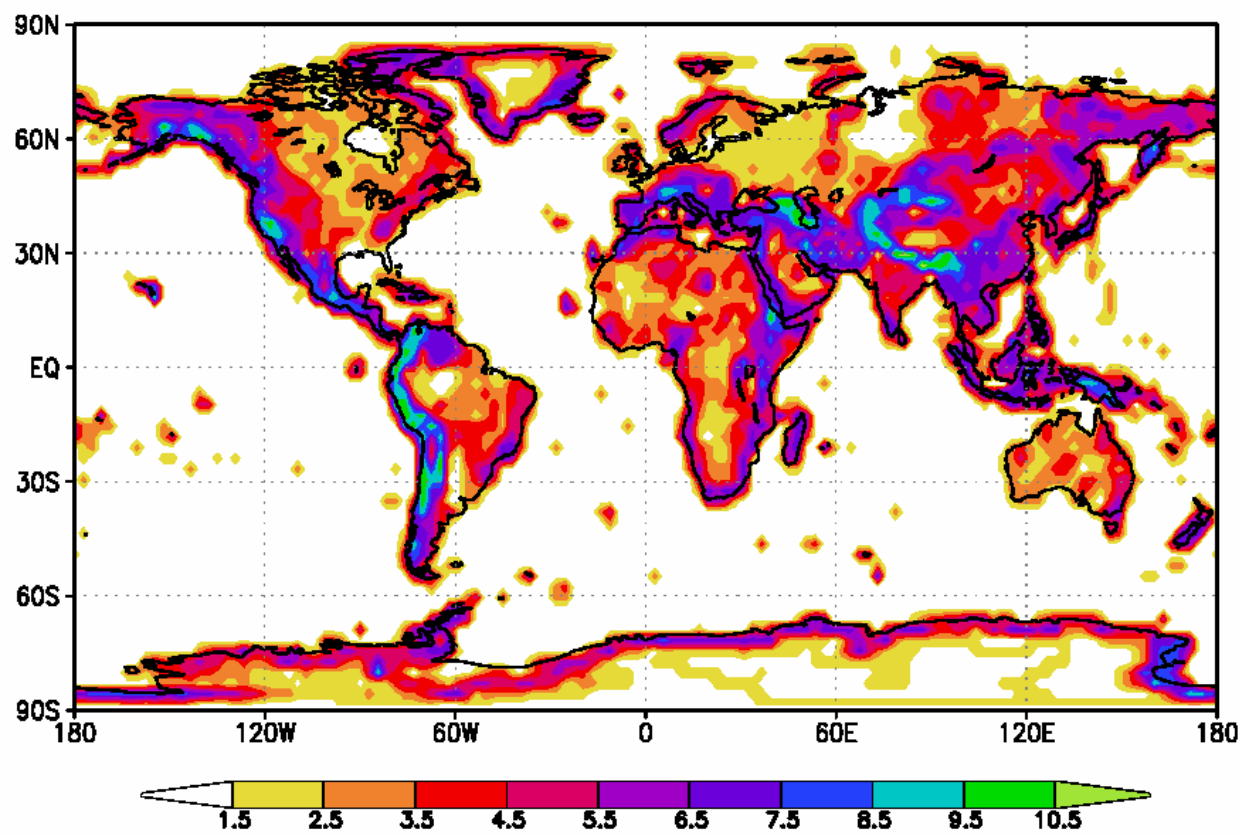


Figure 3. Number of elevation classes within each 2.8ºx2.8ºgrid cell for an eleven-class classification of surface elevation scheme.

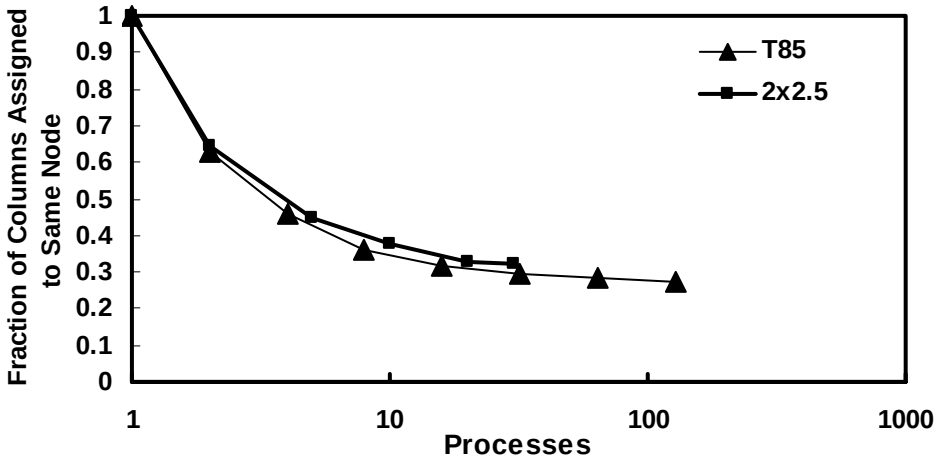
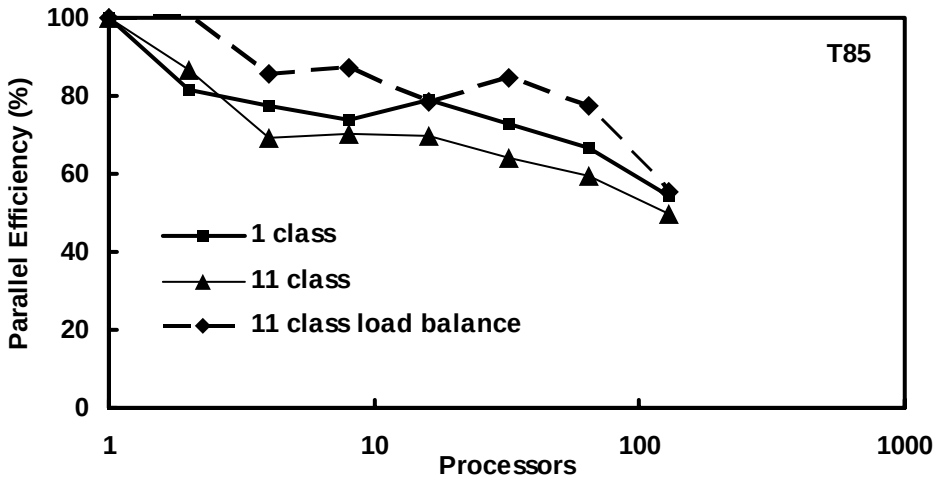


Figure 4. Fraction of physics columns assigned to the same process as the corresponding grid cell for dynamics for T85 spectral resolution and 2ºx2.5º finite-volume resolution.



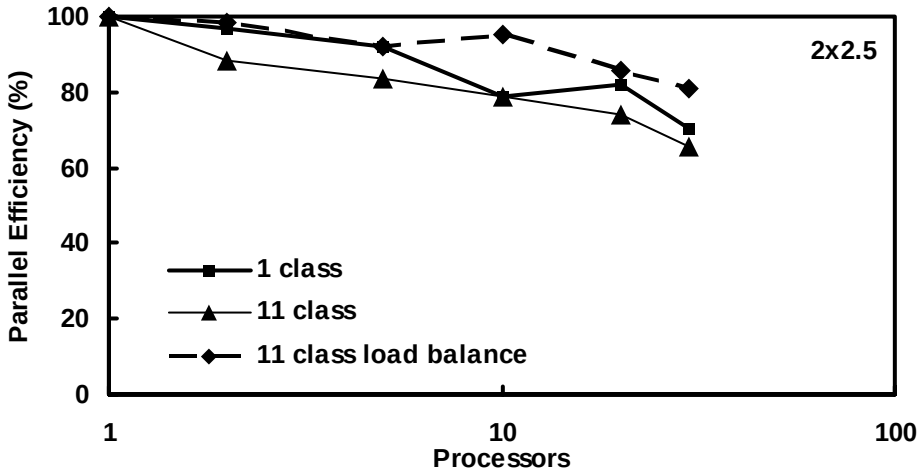


Figure 5. Parallel efficiency as a function of processor count for T85 (above) and 2x2.5 (below) configurations without the subgrid scheme (1 class), with it but without interprocess load balancing (11 class), and with the subgrid scheme and with interprocess load balancing (11class load balance).

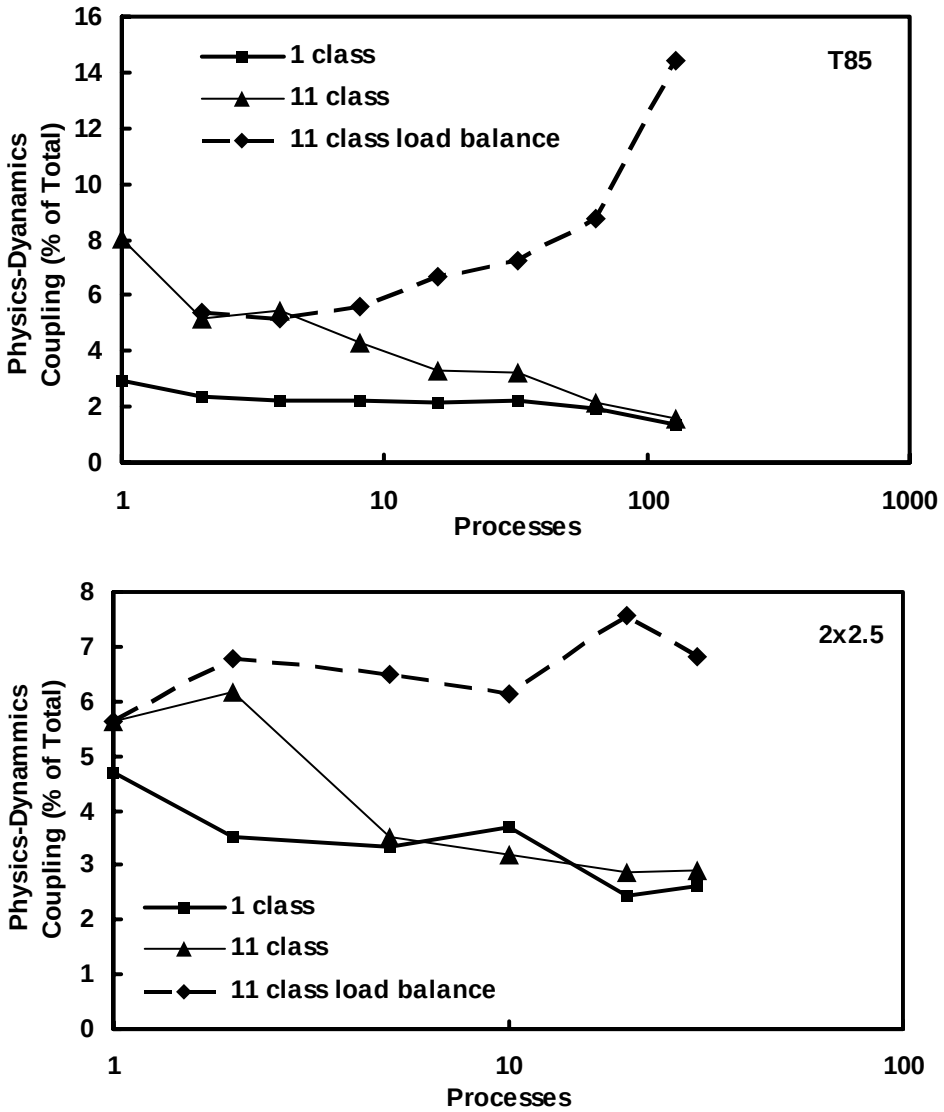


Figure 6. The physics-dynamics coupling time normalized by the total time, as a function of process count for T85 (above) and 2˚x2.5˚ (below) resolution for simulations with (11 class) and without (1 class) the subgrid scheme with and without load balancing.

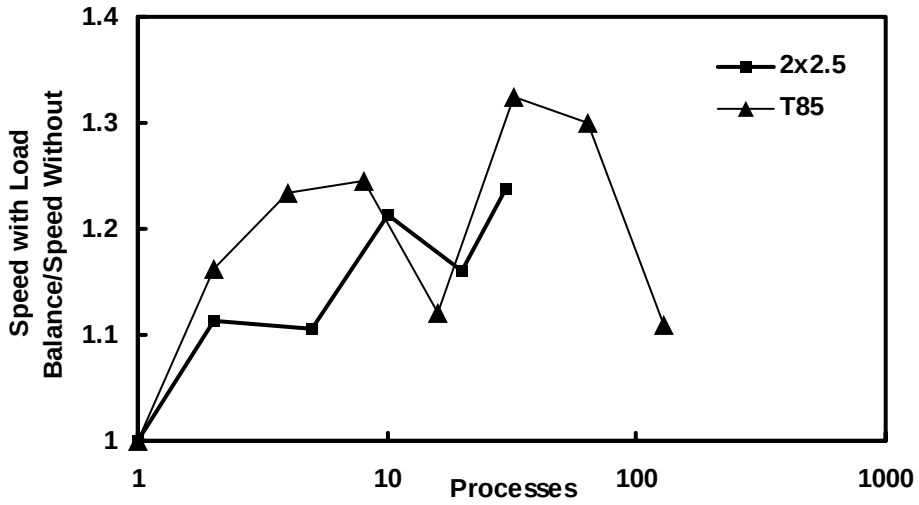


Figure 7. The simulation speed with load balancing normalized by the speed without load balancing as functions of process count for T85 (above) and 2˚x2.5˚ (below) resolution.

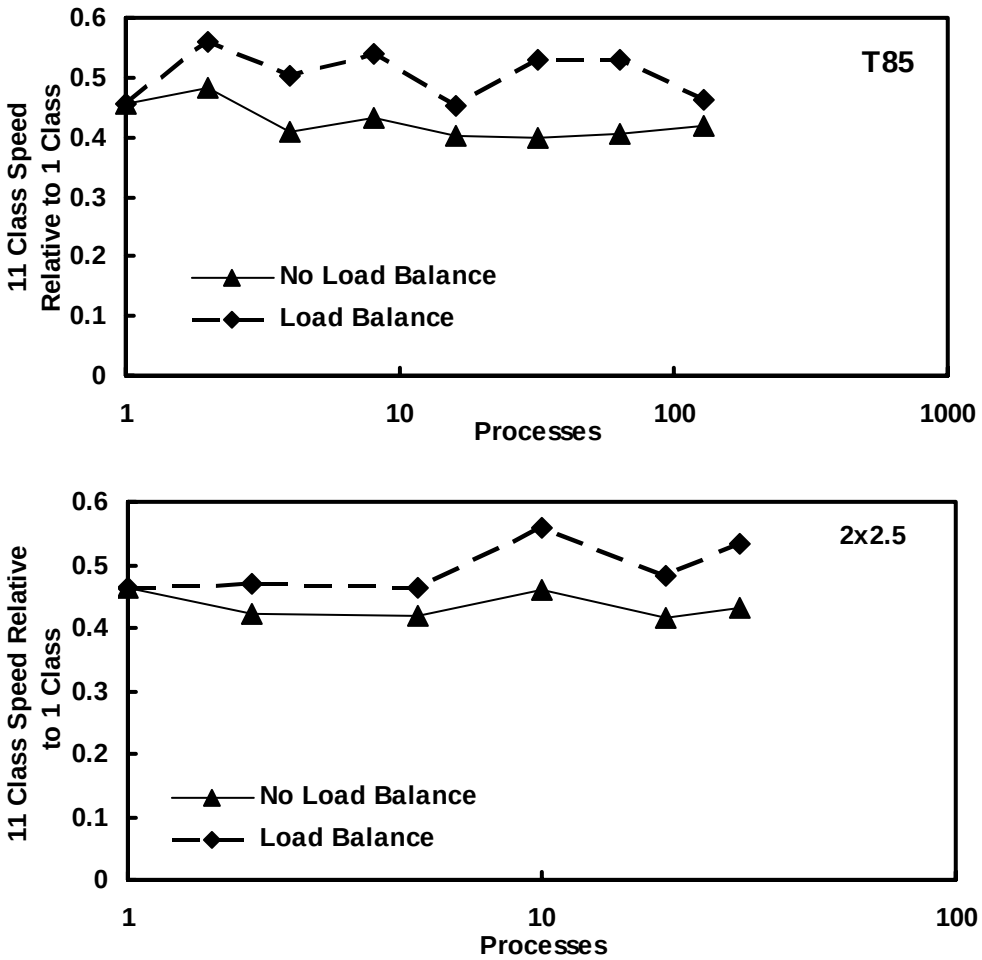


Figure 8. The CAM3 simulation speed with the subgrid scheme with and without load balancing, both normalized by the simulation speed without the subgrid scheme, for T85 (above) and 2˚x2.5˚ (below) resolution.

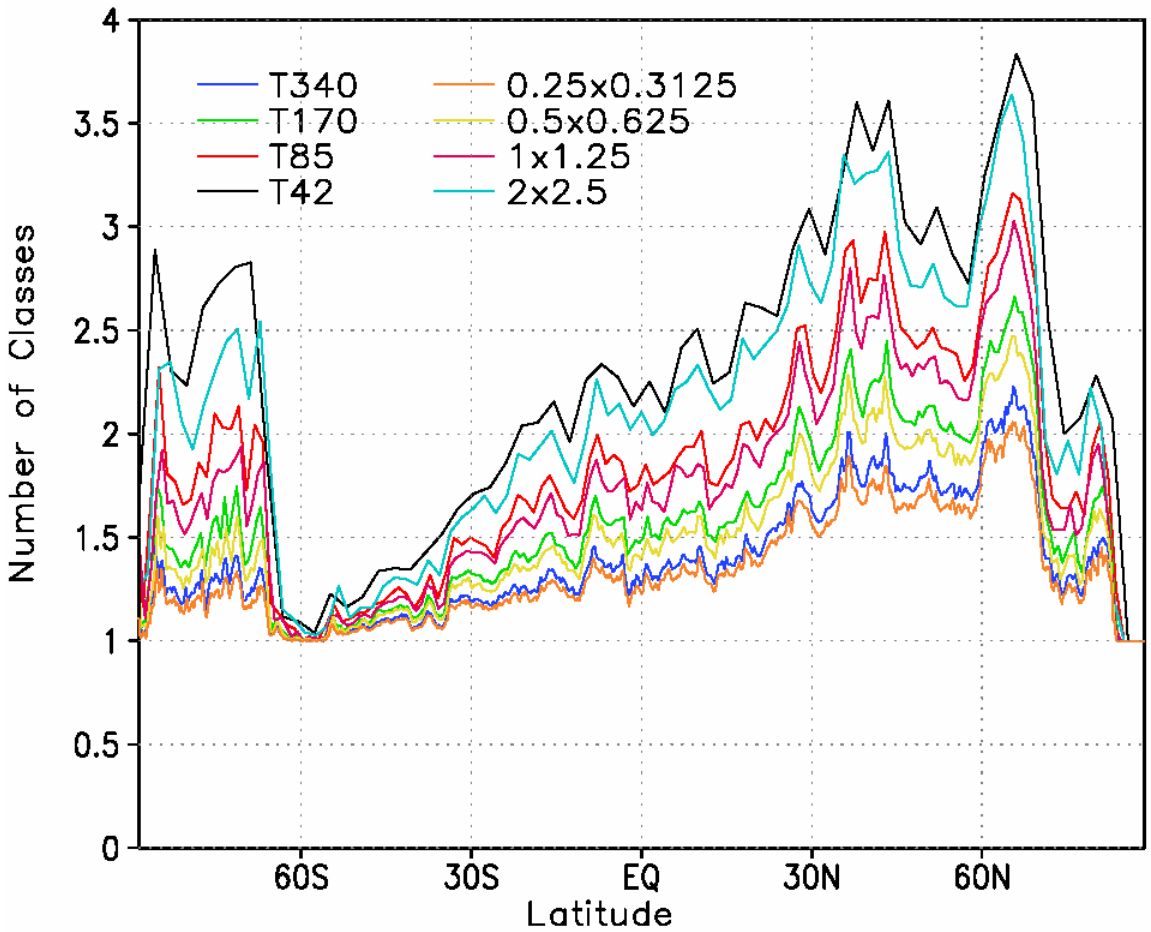


Figure 9. The number of elevation classes per grid cell averaged over longitude for a variety of spectral and finite-volume resolutions, using the eleven-class classification scheme of Ghan et al. (2002).