

First Code Improvement Report for the  
Land Information System  
Submitted under Task Agreement GSFC-CT-2  
Cooperative Agreement Notice (CAN)  
CAN-00OES-01  
Increasing Interoperability and Performance of  
Grand Challenge  
Applications in the Earth, Space, Life, and  
Microgravity Sciences

Version 1.0

History:

Revision	Summary of Changes	Date
1.0	Draft	04/01/03

## Contents

<b>1</b>	<b>Description of the Milestone</b>	<b>4</b>
<b>2</b>	<b>Description of Algorithms</b>	<b>4</b>
2.1	Land Surface Modeling and Data Assimilation . . . . .	4
2.2	LIS driver . . . . .	4
2.3	Community Land Model (CLM) . . . . .	7
2.4	The Community NOAA Land Surface Model . . . . .	7
2.5	Code Improvements and Parallelization in LIS . . . . .	8
<b>3</b>	<b>Description of the Test Case</b>	<b>9</b>
<b>4</b>	<b>Description of the Computer Code Used</b>	<b>10</b>
4.1	Documentation of the Computer Code . . . . .	11
4.2	Code Repository . . . . .	11
<b>5</b>	<b>Results</b>	<b>11</b>
5.1	Results at 1/4° domain . . . . .	11
5.1.1	Profiling Results . . . . .	11
5.1.2	Execution Times and Scaling . . . . .	12
5.1.3	Memory Usage . . . . .	13
5.2	Results at 5km . . . . .	16
5.2.1	Execution Times and Scaling . . . . .	16
5.2.2	Memory and Disk Storage . . . . .	17
<b>6</b>	<b>Final Remarks</b>	<b>17</b>

## List of Figures

1	Flowchart for LIS driver . . . . .	6
2	Parallelization Scheme in LIS . . . . .	10
3	Total times for computationally intensive functions on Lomax (from Milestone E) . . . . .	12
4	Total times for computationally intensive functions on Lomax . . . . .	13
5	Percentage of total times for computationally intensive functions on Lomax . . . . .	13
6	Timing Results on Lomax at 1/4° . . . . .	14
7	Timing Results on Chapman 1/4° . . . . .	14
8	Comparison of Timing Results on Lomax and Chapman . . . . .	15

9 Timing Results on Lomax at 5km . . . . . 17

## List of Tables

1 Measure of computational intensity for LIS driver at  $1/4^\circ$  (ms/gridcell/day) 15  
2 Memory Usage for various LIS driver runs . . . . . 16  
3 Measure of computational intensity for LIS driver at 5km (ms/gridcell/day) 16

## 1 Description of the Milestone

The milestone for the Land Information System (LIS) [5] code baseline deals with the implementation and execution of the Community Land Model (CLM) [2] and the National Oceanic and Atmospheric Administration's NOAA (National Center for Environmental Prediction, Oregon State University, United States Air Force, and Office of Hydrology) [6] land surface model (LSM) within the LIS driver at 5km resolution on the ESS Testbed for the near-term retrospective period running at approximately 1ms/gridcell/day. The milestone also requires publishing an initial version of documented source code made publicly available via the Web. The expected completion date is March 2003.

## 2 Description of Algorithms

This section provides an overall description of the land surface modeling and data assimilation, followed by a description of the algorithms for each individual components of LIS involved in this code improvement study.

### 2.1 Land Surface Modeling and Data Assimilation

In general, land surface modeling seeks to predict the terrestrial water, energy and biogeochemical processes by solving the governing equations of the soil-vegetation-snowpack medium. Land surface data assimilation seeks to synthesize data and land surface models to improve our ability to predict and understand these processes. The ability to predict terrestrial water, energy and biogeochemical processes is critical for applications in weather and climate prediction, agricultural forecasting, water resources management, hazard mitigation and mobility assessment.

In order to predict water, energy and biogeochemical processes using (typically 1-D vertical) partial differential equations, land surface models require three types of inputs: 1) initial conditions, which describe the initial state of land surface; 2) boundary conditions, which describe both the upper (atmospheric) fluxes or states also known as "forcings" and the lower (soil) fluxes or states; and 3) parameters, which are a function of soil, vegetation, topography, etc., and are used to solve the governing equations.

### 2.2 LIS driver

The main driver in LIS is derived from the Land Data Assimilation System (LDAS) [5]. LDAS is a model control and input/output system (consisting of a number of sub-

routines, modules written in Fortran 90 source code) that drives multiple offline one dimensional land surface models (LSMs) using a vegetation defined "tile" or "patch" approach to simulate sub-grid scale variability. The one-dimensional LSMs such as CLM and NOAH, which are subroutines of LDAS, apply the governing equations of the physical processes of the soil-vegetation-snowpack medium. These land surface models aim to characterize the transfer of mass, energy, and momentum between a vegetated surface and the atmosphere.

LDAS makes use of various satellite and ground based observation systems within a land data assimilation framework to produce optimal output fields of land surface states and fluxes. The LSM predictions are greatly improved through the use of a data assimilation environment such as the one provided by LDAS. In addition to being forced with real time output from numerical prediction models and satellite and radar precipitation measurements, LDAS derives model parameters from existing topography, vegetation and soil coverages. The model results are aggregated to various temporal and spatial scales, e.g., 3 hourly,  $1/4^\circ$ . The LDAS driver was used in the baselining results presented for Milestone E. The LIS driver used for demonstrating code improvements for Milestone H was developed by adopting the core LDAS driver and implementing code improvements for enhancing performance. The structure of LDAS driver was also redesigned using object oriented principles, providing adaptable interfaces for ease of code development and extensibility. For a detailed description of the redesign and code improvements, please refer to the interoperability document.

Figure 1 shows the algorithmic steps involved in the LIS driver. The execution of LIS driver starts with reading in the user specifications. The user selects the model domain and spatial resolution, the duration and timestep of the run, the land surface model, the type of forcing from a list of model and observation based data sources, the number of "tiles" per grid square (described below), the soil parameterization scheme, reading and writing of restart files, output specifications, and the functioning of several other enhancements including elevation correction and data assimilation.

The system then reads the vegetation information and assigns subgrid tiles on which to run the one-dimensional simulations. LIS driver runs its 1-D land models on vegetation-based "tiles" to simulate variability below the scale of the model grid squares. A tile is not tied to a specific location within the grid square. Each tile represents the area covered by a given vegetation type.

Memory is dynamically allocated to the global variables, many of which exist within Fortran 90 modules. The model parameters are read and computed next. The time loop begins and forcing data is read, time/space interpolation is computed and modified as necessary. Forcing data is used to specify boundary conditions to the land surface model. The LSMs in the LIS driver are driven by atmospheric forcing data such as precipitation, radiation, wind speed, temperature, humidity, etc., from various sources. LIS driver applies spatial interpolation to convert forcing data to

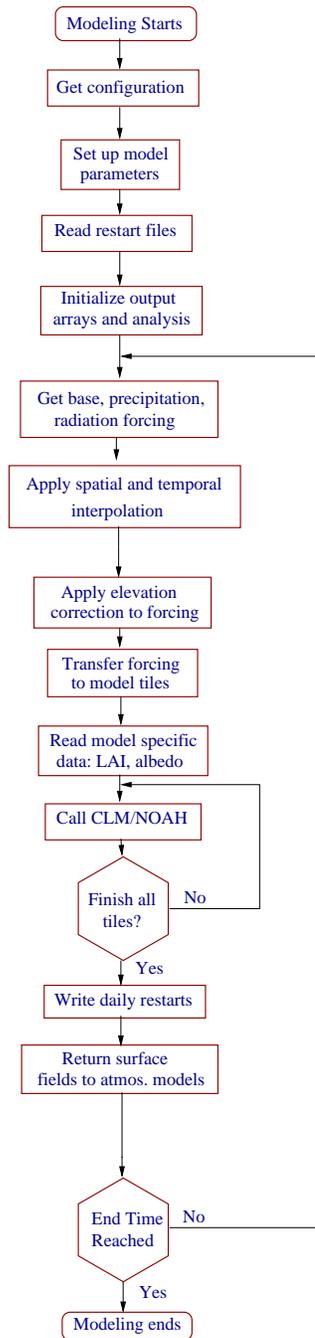


Figure 1: Flowchart for LIS driver

the appropriate resolution required by the model. Since the forcing data is read in

at certain regular intervals, LIS driver also temporally interpolates time average or instantaneous data to that needed by the model at the current timestep. The selected model is run for a vector of “tiles”, intermediate information is stored in modular arrays, and output and restart files are written at the specified output interval.

### 2.3 Community Land Model (CLM)

CLM is a 1-D land surface model, written in Fortran 90, developed by a grass-roots collaboration of scientists who have an interest in making a general land model available for public use. LIS currently uses CLM version 2.0, which was released in May 2002. The source code for CLM 2.0 is freely available from the National Center for Atmospheric Research (NCAR) (<http://www.cgd.ucar.edu/tss/clm/>). CLM is used as the land model for the community climate system model (CCSM) (<http://www.cesm.ucar.edu/>) and the community atmosphere model (CAM) (<http://www.cgd.ucar.edu/cms/>). CLM is executed with all forcing, parameters, dimensioning, output routines, and coupling performed by an external driver of the user’s design (in this case done by LIS driver). CLM requires pre-processed data such as the land surface type, soil and vegetation parameters, model initialization, and atmospheric boundary conditions as input. The model applies finite-difference spatial discretization methods and a fully implicit time-integration scheme to numerically integrate the governing equations. The model subroutines apply the governing equations of the physical processes of the soil-vegetation-snowpack medium, including the surface energy balance equation, Richards’ [7] equation for soil hydraulics, the diffusion equation for soil heat transfer, the energy-mass balance equation for the snowpack, and the Collatz et al. [3] formulation for the conductance of canopy transpiration.

### 2.4 The Community NOAA Land Surface Model

The community NOAA Land Surface Model is a stand-alone, uncoupled, 1-D column model freely available at the National Centers for Environmental Prediction (NCEP; <ftp://ftp.ncep.noaa.gov/pub/gcp/ldas/noah1sm/>). NOAA can be executed in either coupled or uncoupled mode. It has been coupled with the operational NCEP mesoscale Eta model [1] and its companion Eta Data Assimilation System (EDAS) [8], and the NCEP global Medium-Range Forecast model (MRF) and its companion Global Data Assimilation System (GDAS). When NOAA is executed in uncoupled mode, near-surface atmospheric forcing data (e.g., precipitation, radiation, wind speed, temperature, humidity) is required as input. NOAA simulates soil moisture (both liquid and frozen), soil temperature, skin temperature, snowpack depth, snowpack water equivalent, canopy water content, and the energy flux and water flux terms of the surface energy balance and surface water balance. The model

applies finite-difference spatial discretization methods and a Crank-Nicholson time-integration scheme to numerically integrate the governing equations of the physical processes of the soil vegetation-snowpack medium, including the surface energy balance equation, Richards' [7] equation for soil hydraulics, the diffusion equation for soil heat transfer, the energy-mass balance equation for the snowpack, and the Jarvis [4] equation for the conductance of canopy transpiration.

## 2.5 Code Improvements and Parallelization in LIS

Our profiling results from Milestone E demonstrated the functions that are most time-consuming, thereby identifying the portions of our code-set that require our immediate attention. These functions were mainly the spatial interpolation, temporal interpolation, and the land surface model runs. The code improvements for Milestone F concentrated on improving the performance of these functions. For spatial interpolation, our code employs an external library called `ipolates` from NCEP. The source code of this library's routines were rewritten to improve their performance. The time interpolation and the land surface model runs were parallelized using the scheme described below.

Most of the community LSMs and the LDAS were not originally designed with an intent to be run at such high resolutions such as 5km. As a result, the internal data structures in these models and driver consisted of large data structures leading to considerable memory requirements at 5km. For example, a "NOAH" tile will be defined for every point in the tilespace. The amount of memory required for the whole NOAH data structure will scale linearly with increase in grid points, leading to unmanageable memory requirements.

Our original estimates of memory from the baselining results predicts approximately 50GB and 90GB of memory for NOAH and CLM runs approximately. Our baselining code used an older version of CLM (Version 1.0). CLM version 2.0 is significantly more complex than version 1.0 in its functionality. For example, CLM version 1.0's and 2.0's main data structures contain approximately 450 and 550 variables, respectively. CLM version 2.0 contains a number of additional functionalities such as capability for coupling with atmosphere models, modules for river routing, volatile organic compounds emissions, Dust mobilization analysis, etc. CLM2.0 includes extensive changes in surface datasets such as Leaf Area Index(LAI), Stem Area Index (SAI), canopy heights, percent clay and percent sand. CLM1.0 uses a single LAI value varying between a prescribed max/min value determined by soil temperature, whereas CLM2.0 uses a geographically and temporally varying value. In terms of the land surface physics, CLM2.0 includes calculation of orbital parameters and the plant functional types determine the vegetation parameters, in addition to the sub-models described above.

The entire code was redesigned, with an emphasis on reducing the size of the main data structures. The constructs in the LIS driver and the land surface model were significantly modified, including the vectorization scheme mentioned below, leading to considerable memory savings.

The global land surface is modeled by dividing it into two-dimensional regions or cells (e.g. cells of size 1km x 1km, which would lead to approximately 50,000 times more grid points than that of a simulation with cells of size  $2^\circ \times 2.5^\circ$ ). Each cell can have a partial spatial coverage by a number of vegetation types, as well as bare soil. The vegetation characteristics such as leaf area index, stomatal resistance, etc. might be time varying. The conditions in each cell (energy, water fluxes, etc.) are computed at different time intervals. Each cell is driven by different atmospheric forcing variables. Assuming approximately 0.4 milliseconds for each LSM run on a particular cell, it can be estimated that modeling land surface processes over a year with 15 minute timesteps would require approximately 74 years of runtime. This problem is clearly a grand challenge simply from computational perspective.

Land surface processes have rather weak horizontal coupling on short time and large space scales, which enables efficient scaling across massively parallel computational resources. LIS driver takes advantage of this weak horizontal coupling of land surface processes in the parallelization scheme described below.

Figure 2 shows the parallelization scheme employed in LIS. The program starts by initializing the global grid and associated parameters. Since land surface modeling is conducted only on the land points, the program switches to a vectorized representation of grid, which in turn leads to significant memory savings. For example, at 5km resolution, the number of grid points is approximately 21 million. By using a vectorized grid representation, we conduct model runs on only approximately 6 million points. A dynamic domain decomposition based on the number of processors is carried out on the vectorized grid. The master processor conducts the spatial interpolation. Temporal interpolation and land surface model runs are carried out on the compute nodes based on their assigned subdomain. The master collects the variables required for output, and the loop continues until the simulation is complete.

### 3 Description of the Test Case

The baselining results presented in this report were obtained by executing the LIS driver on the following NASA AMES systems.

- LOMAX: SGI Origin 3000 IRIX64 6.5, 512 400MHz IP35 Processors
- CHAPMAN: SGI Origin 3000 IRIX64 6.5, 1024 600MHz IP35 Processors

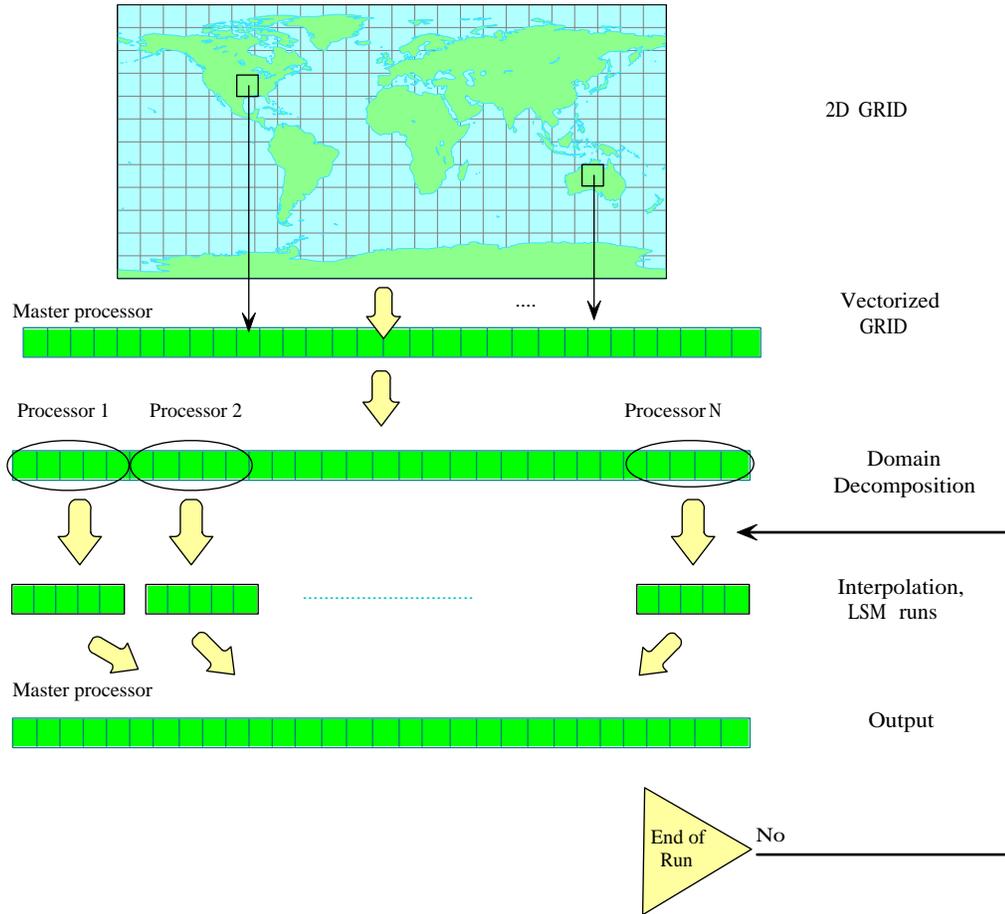


Figure 2: Parallelization Scheme in LIS

The domain resolution was set to be 5km. There are 21,600,000 grid points at 5km resolution. The CLM and NOAH LSMs were used in various runs. A timestep of 30 minutes was used in the runs. For simplicity, only one tile per grid was simulated in the runs. The output files were written using binary format. The GEOS base forcing was employed in the runs. The scalability of the code at different domain resolutions was also examined.

## 4 Description of the Computer Code Used

This section provides an algorithmic description of the computer code used in the baselining. LIS driver is a model control and input/output system (consisting of

a number of subroutines, modules written in Fortran 90 source code) that drives multiple offline one-dimensional LSMs using a vegetation defined “tile” or “patch” approach to simulate subgrid scale variability. The one-dimensional LSMs, which are subroutines of LIS driver, apply the governing equations of the physical processes of the soil-vegetation-snowpack medium. These equations are model independent.

## 4.1 Documentation of the Computer Code

The documentation of the LIS driver and the land surface models (CLM 2.0 and NOAH 2.5) can be accessed at <http://lis.gsfc.nasa.gov/Documentation/MilestoneF/Documentation/index.html>

## 4.2 Code Repository

The computer source code employed for this code improvement may be obtained from the LIS source code repository at <http://lis2.sci.gsfc.nasa.gov:9090/Fcode/>.

# 5 Results

The LIS driver was run on different SGI Origin systems using GEOS forcing and different land surface models. The simulated period of time considered for all the runs in this study is 1 day. The computational demands of various runs are quantified using four parameters: Total execution times, CPU times, disk usage, and memory usage. We present the computational results obtained at  $1/4^\circ$  as well as at 5km for comparison.

## 5.1 Results at $1/4^\circ$ domain

The computational performance results of the improved code at  $1/4^\circ$  is presented in this section for comparison with the baselining results presented for Milestone E.

### 5.1.1 Profiling Results

A dynamic runtime profiling, using SGI’s speedshop toolkit, was conducted to identify the computationally intensive segments of the improved code. This profiling was conducted running the code on a single processor. Figures 4 and 5 shows the computational times and the percentage of total times, respectively, of the time consuming functions. Figure 3 shows the corresponding profiling results from Milestone E. It can be observed that the computational requirements of the spatial interpolation routines

are significantly reduced from the baselining results of Milestone E. The time consuming functions in the improved code is the time interpolation and land surface model runs. The increase in the land surface model executions can be associated with the added functionalities. It can also be observed that the computational requirements of the CLM main driver is significantly more than that of NOAH.

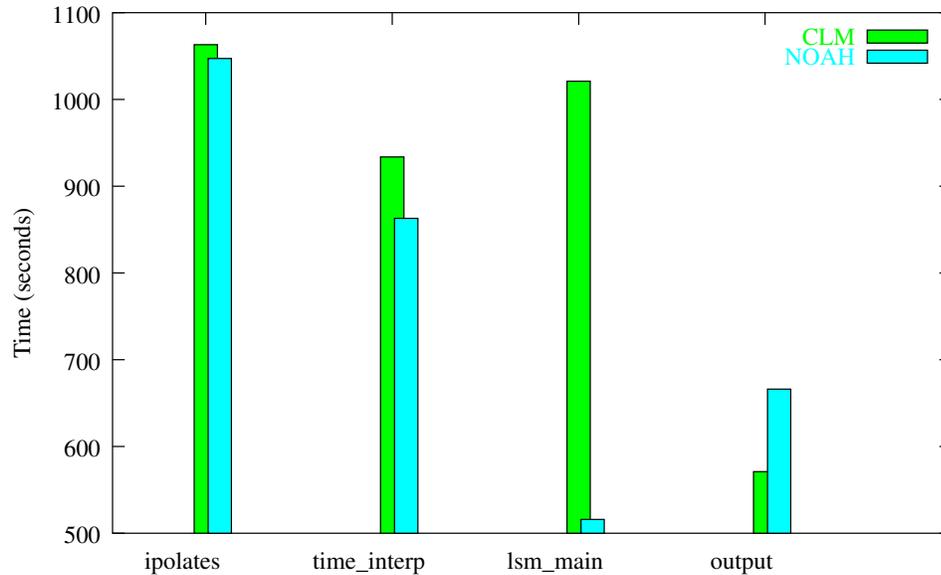


Figure 3: Total times for computationally intensive functions on Lomax (from Milestone E)

### 5.1.2 Execution Times and Scaling

Figures 6 and 7 shows the scaling curves of the improved code at  $1/4^\circ$  for both Lomax and Chapman. The code performs better than that of  $P/2$  scaling for runs upto 16 processors. Figure 8 shows the comparison of runs on Chapman and Lomax. The scaling behavior is similar on both platforms and code performs better on Chapman, as expected.

Table 1 shows the computational intensity values for  $1/4^\circ$  runs on Chapman and Lomax. The improved code performance for the sequential run is significantly better than that of the baseline code. For example, on Lomax, the NOAH computational intensity for the baseline code was approximately 4.4 and that of CLM was approximately 7.7. Even with the more complex CLM2 model, the improved code performs much better than the baseline code.

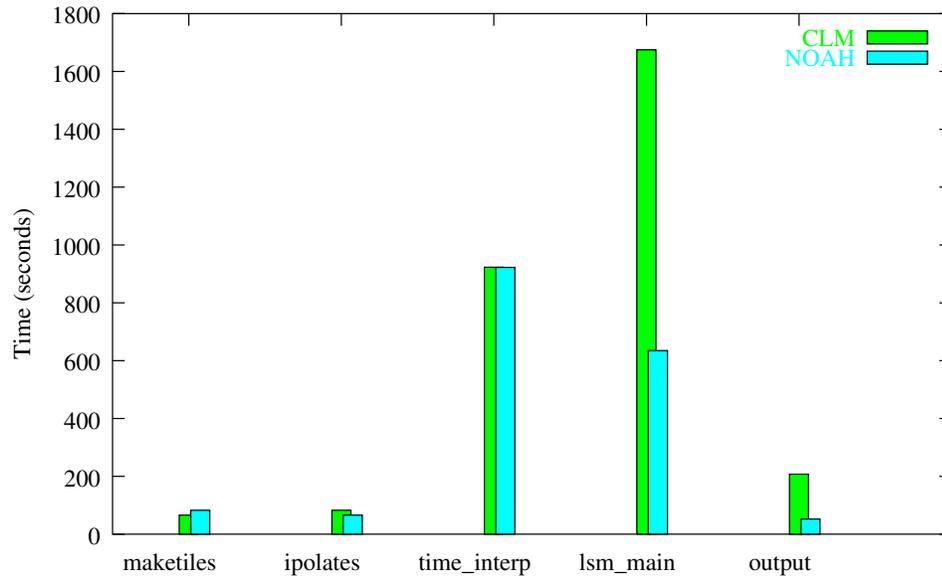


Figure 4: Total times for computationally intensive functions on Lomax

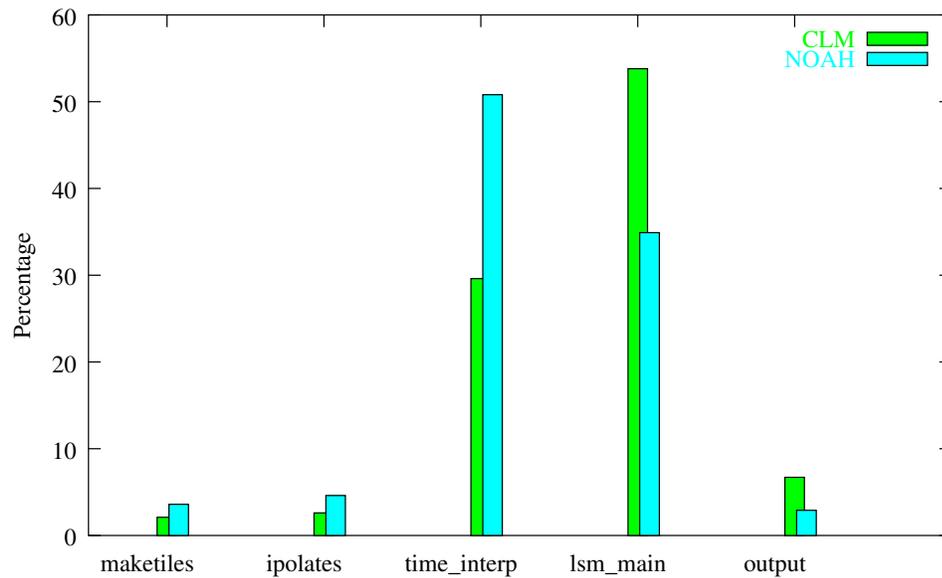


Figure 5: Percentage of total times for computationally intensive functions on Lomax

### 5.1.3 Memory Usage

The LIS driver code also requires significant memory for execution. The following Table 2 lists the approximate memory requirements for runs with different land sur-

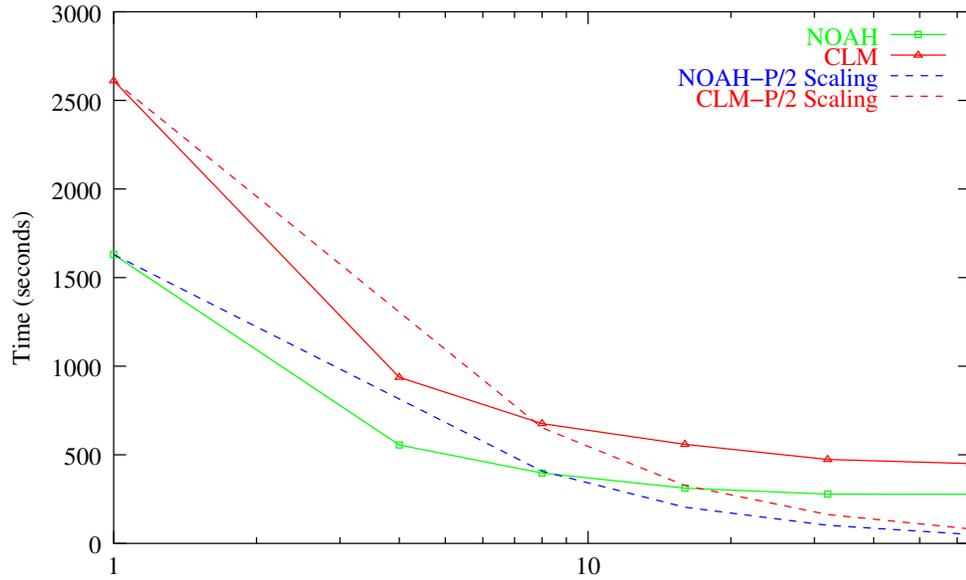


Figure 6: Timing Results on Lomax at 1/4°

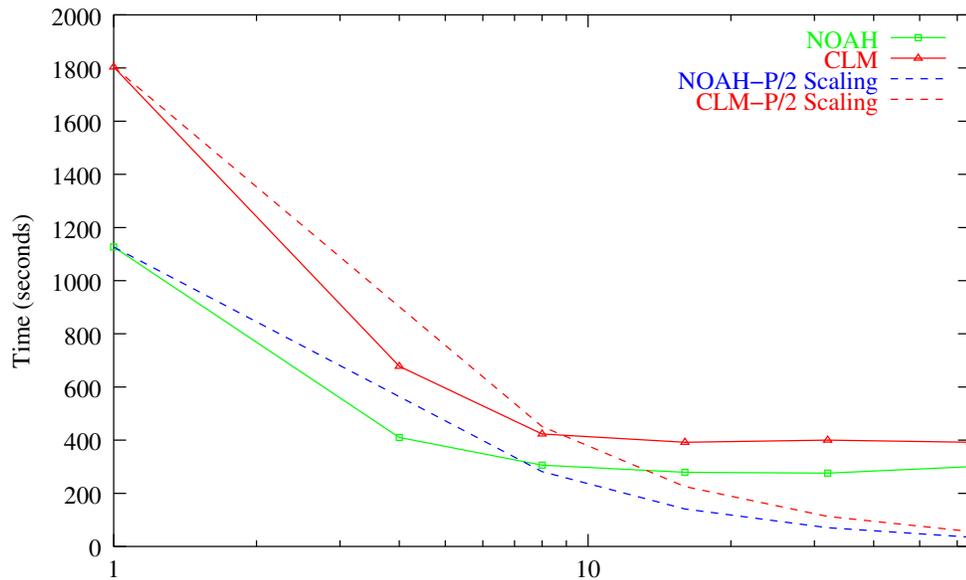


Figure 7: Timing Results on Chapman 1/4°

face models. These numbers are obtained from the memory profiling conducted on Chapman. The improved code demonstrates significant memory savings as expected from our improvements described earlier.

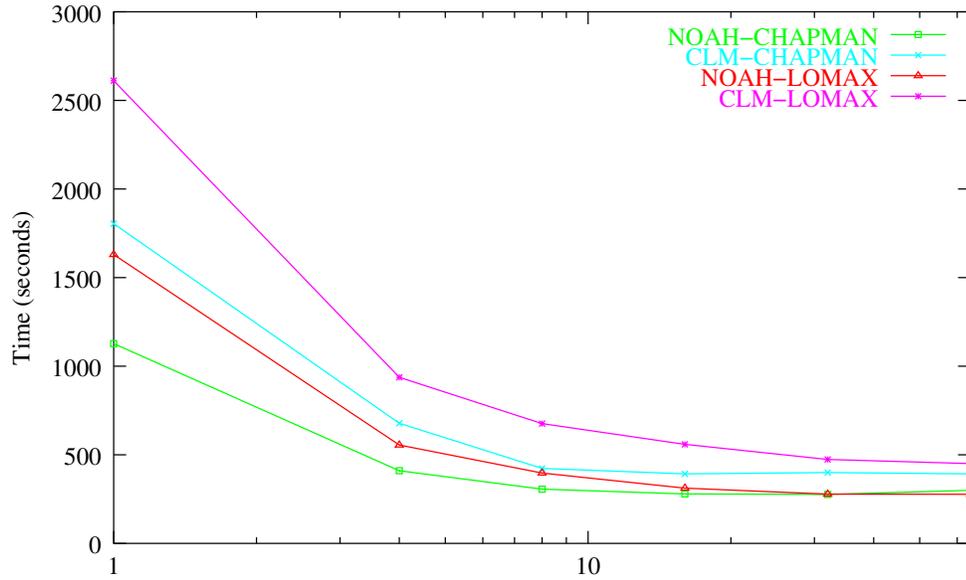


Figure 8: Comparison of Timing Results on Lomax and Chapman

Table 1: Measure of computational intensity for LIS driver at  $1/4^\circ$  (ms/gridcell/day)

	Number of Processors	CLM Timestep (minutes)	NOAH Timestep (minutes)
		30	30
Lomax	1	3.02	1.89
	4	1.08	0.64
	8	0.78	0.46
	16	0.65	0.36
	32	0.55	0.32
	64	0.52	0.32
Chapman	1	2.08	1.3
	4	0.78	0.47
	8	0.49	0.35
	16	0.45	0.33
	32	0.46	0.32
	64	0.45	0.34

Table 2: Memory Usage for various LIS driver runs

	CLM	NOAH
LDAS 1/4° (Milestone E)	3.5 GB	1.4 GB
LIS driver 1/4° (Milestone F)	0.95 GB	0.50 GB

## 5.2 Results at 5km

The computational performance results of the improved code at 5km resolution is presented in this section.

### 5.2.1 Execution Times and Scaling

5km land surface model simulations were conducted using both the CLM and NOAH models on Lomax. Figure 9 presents the scaling curves for both these models at 5km. The sequential executions of these models for a day’s simulation require approximately 18 and 12 hours for CLM and NOAH, respectively. By taking advantage of the parallelization scheme, they require approximately 3 and 2 hours, respectively. It can be seen that the scaling behavior at 5km is similar to that observed at 1/4°.

Table 3 lists the computational intensity values for the 5km runs, for a varying the number of processors. These normalized values are close to those calculated for the 1/4°. The target of 1ms/gridcell/day is achieved by using 8 processors, and the performance is further improved by increasing the number of processors.

Table 3: Measure of computational intensity for LIS driver at 5km (ms/gridcell/day)

	Number of Processors	CLM Timestep (minutes)	NOAH Timestep (minutes)
Lomax	1	2.96	1.95
	8	0.95	0.48
	16	0.68	0.39
	32	0.64	0.34
	64	0.57	0.32
	128	0.51	0.30

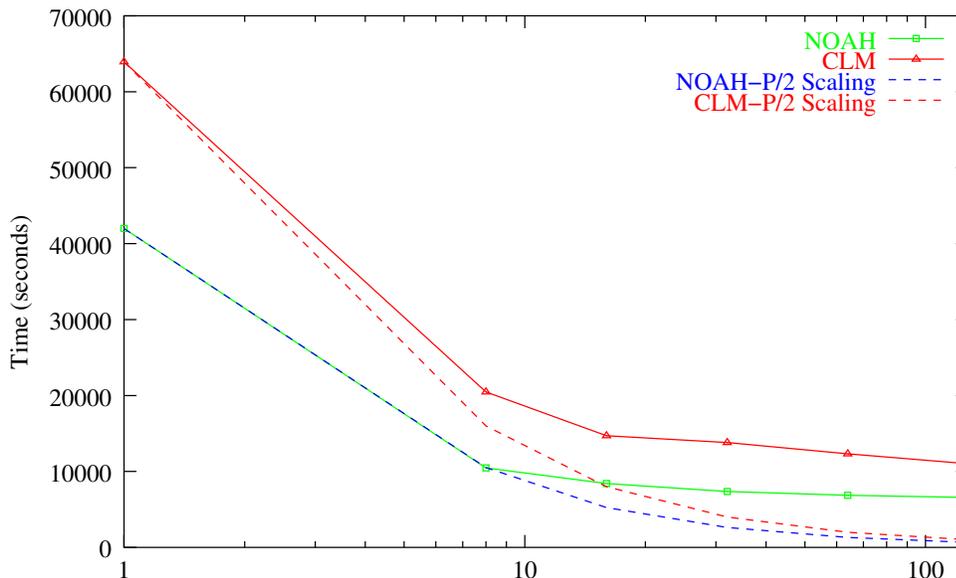


Figure 9: Timing Results on Lomax at 5km

### 5.2.2 Memory and Disk Storage

The memory required for 5km simulations significantly increases at 5km. CLM execution at 5km requires approximately 27GB and NOAH requires approximately 12GB. This is consistent with our linear extrapolation from  $1/4^\circ$ .

The output files generated by the 5km run was approximately 1.4 GB, every 3 hours. This is also consistent with our estimate using the  $1/4^\circ$  run, which produces output sizes of approximately 54MB, every 3 hours. For CLM, the output sizes are approximately 71 MB and 1.7GB, for  $1/4^\circ$  and 5km, respectively.

## 6 Final Remarks

The scaling results presented for both  $1/4^\circ$  and 5km domain shows the computational improvements in using the parallelization scheme. It can also be noted that the computational intensity values are significantly less than the target 1ms/gridcell/day and are close to the target 0.4ms/gridcell/day, when the simulations need to be carried out at 1km.

Our memory profiling results show the improvements we made by redesigning the drivers and the land surface models. However, our 5km runs still require considerable memory that limits us currently to a shared memory architecture that provides adequate memory. In order to port our code to a limited memory architecture such

as a Linux cluster, we need to redesign our data hosting and IO so that the IO of global files and output can be handled in chunks. The next code improvement will focus on the redesign of IO so that LIS code can be ported to a Linux cluster. The IO redesign will also focus on managing the huge output volume, especially at 1km.

## References

- [1] F. Chen, K. Mitchell, J. Schaake, Y. Xue, H. Pan, V. Koren, Y. Duan, M. Ek, and A. Betts. Modeling of land-surface evaporation by four schemes and comparison with fife observations. *J. Geophys. Res.*, 101(D3):7251–7268, 1996.
- [2] CLM. <http://www.cgd.ucar.edu/tss/clm/>.
- [3] G. J. Collatz, C. Grivet, J. T. Ball, and J. A. Berry. Physiological and environmental regulation of stomatal conductance: Photosynthesis and transpiration: A model that includes a laminar boundary layer. *Agric. For. Meteorol.*, 5:107–136, 1991.
- [4] P. G. Jarvis. The interpretation of leaf water potential and stomatal conductance found in canopies of the field. *Phil. Trans. R. Soc.*, B(273):593–610, 1976.
- [5] LDAS. <http://ldas.gsfc.nasa.gov>.
- [6] NOAH. <ftp://ftp.ncep.noaa.gov/pub/gcp/ldas/noahsm/>.
- [7] L. A. Richards. Capillary conduction of liquids in porous media. *Physics*, 1:318–333, 1931.
- [8] E. Rogers, T. L. Black, D. G. Deaven, G. J. DiMego, Q. Zhao, M. Baldwin, N. W. Junker, and Y. Lin. Changes to the operational "early" eta analysis/forecast system at the national centers of environmental prediction. *Wea. Forecasting*, 11:391–413, 1996.