Best, B., J., Furjanic, C., Gerhart, N., Fincham, J., Gluck, K. A., Gunzelmann, G., & Krusmark, M. A. (2009). Adaptive mesh refinement for efficient exploration of cognitive architectures and cognitive models. In A. Howes, D. Peebles, & R. Cooper (Eds.). *Proceedings of the Ninth International Conference on Cognitive Modeling*. Manchester, United Kingdom: University of Manchester.

Adaptive Mesh Refinement for Efficient Exploration of Cognitive Architectures and Cognitive Models

Bradley J. Best (bjbest@AdCogSys.com)
Caitlin Furjanic (cfurjanic@AdCogSys.com)
Nathan Gerhart (ngerhart@AdCogSys.com)
Adaptive Cognitive Systems, 1942 Broadway St., Suite 201
Boulder, CO, 80302 USA

Jon Fincham (Fincham@cmu.edu)
Department of Psychology, 5000 Forbes Ave.
Pittsburgh, PA, 15213 USA

Kevin A. Gluck (kevin.gluck@us.af.mil)
Glenn Gunzelmann (glenn.gunzelmann@us.af.mil**)**
Air Force Research Laboratory, 6030 S. Kent St.
Mesa, AZ, 85212 USA

Michael A. Krusmark (michael.krusmark@mesa.afmc.af.mil)
L-3 Communications, 6030 S. Kent St.
Mesa, AZ, 85212 USA

Abstract

The majority of cognitive models support some form of parameterization, either of the model itself, or through architectural mechanisms. In order to fully understand these models, it is important to understand the model's behavior as a result of parameter variation across a wide range of values. Even simple models become difficult to understand without a systematic method of exploring performance across parameter combinations, and scientists have turned to iterative methods to perform sweeps of these spaces. As an alternative to an exhaustive, homogeneous search, we examined adaptive mesh refinement (AMR) to explore simple and complex parameter spaces of several models developed within ACT-R. AMR allows for fewer model runs with minimal loss of information. We found that, with appropriate granularity, AMR methods can provide a sufficient computational exploration of a performance space with only 1% of the sampling of conventional, homogeneous parameter sweeps. The advantages of AMR for computationally efficient exploration of the performance predictions should be of benefit and interest to developers and users of cognitive architectures and cognitive models.

Keywords: Adaptive mesh refinement; Cognitive architecture; Cognitive model; ACT-R; Parameter sweeps; visualization

Introduction

Although many discussions of cognitive modeling focus on the degree of fit to human empirical data, the point has been compellingly made that what a cognitive model does outside of the best-fitting parameter combination is just as important as what it does at the best-fitting parameter combination, and perhaps even more so (Roberts & Pashler, 2000). Information about how a model performs outside the best-fitting parameter combination provides modelers with information about how likely it is that other parameter combinations result in a comparable fit. It also gives modelers information about the full range of behavior possible from the model and how different parameters interact to generate possibly complex behavioral dynamics. Both novice users of a cognitive architecture working to understand model dynamics, and expert users of a cognitive architecture testing modifications to the theories embedded in these architectures would stand to benefit enormously from a rapid analysis and visualization of the model performance spaces involved. However, cognitive modelers facing this problem are currently confronted with a lack of tools that support exploring that space. The de-facto approach to cognitive modeling is more often a focus on maximizing fit to human data. This is done through either hand-tuning based on the intuition and experience of the modeler or automated optimizing of the fit of a cognitive model through approaches such as genetic algorithms, the conjugate gradient methods, or any of a variety of other alternatives for optimization. Any of these approaches can be sufficiently successful, but they provide little data about the performance of the model outside of the ultimate parameter values used in presenting the final fit.

Cognitive modelers need techniques and tools to support the rapid exploration of parameter spaces in pursuit of understanding of both models and architectures, including methods that support visualization of complex spaces that illuminate model and architecture behavior in response to changes in parameters. We will describe an integrated approach to these explorations that we have developed across our previous research efforts (e.g., Best, Fincham, Gluck, Gunzelmann, & Krusmark, 2008). First, however,

we will turn to a discussion of exploring parameter spaces in the context of cognitive modeling.

Our goal, in this case, is to understand how the architecture and model behave generally and at the best fitting point itself. To get a full understanding of how a model is behaving outside of the best-fitting parameter combination, one approach is to define the limits and step-sizes of a parameter space and then run a model some number of times at each parameter combination (an exhaustive, homogeneous search), where the selected number of runs is intended to provide convergence on the underlying prediction of the model and architecture. This method produces an evenly sampled space that describes the overall behavior of the model. However, resources (time, computation) are allocated evenly between informative and uninformative areas of the space. Informative areas are rich in detail relating the performance of the model or architecture to the underlying parameters. Uninformative (or less informative) areas of the space may take on a variety of different characteristics, such as a degenerate part of the space where a model produces no responses at all. The resources spent on uninformative areas are essentially wasted, as they provide little additional information. Furthermore, a reduction in granularity (step size) can result in oversampling of the parameter space; resources are wasted in this case as well. Even worse, if the model is a preliminary version or prototype, significant effort could be expended exploring a space that could quickly be deemed uninteresting (e.g., a model with a bug that produces spurious results). Adaptive mesh refinement is one technique that can be used to circumvent these issues and focus resources on high information value areas of the model and architecture space.

Adaptive Mesh Refinement

Adaptive mesh refinement (AMR) is a method that can differentially and intelligently allocate resources to areas of a parameter space that call for finer resolution in the modeling based on the presence of more local complexity (Plewa et al. 2005). Briefly, the entire n-dimensional parameter space, which is defined using some set of finite bounds, is initially divided into geometrically regular cells at a very coarse level. The value of each dependent measure the model produces at the midpoint of each cell is estimated based on the previously sampled value of the dependent measures produced at the corners. This estimated or expected value is then compared to the actual value sampled at the midpoint. If the expected and observed values at the midpoint are closer than a predetermined deviation threshold, changes in the dependent measure are estimated to change linearly across the parameter range within the cell, and the dependent values for all target parameter combinations within that cell are populated with linear interpolation based on the sampled corners and midpoint. Alternatively, if the difference between the estimated and measured values for the dependent measure(s) exceeds the threshold, the cell is divided more finely and the process is repeated with the children cells. Ultimately, this results in minimal sampling over linear portions of the space and maximal (bounded) sampling over areas that have more complex surface characteristics (e.g., curvature, variability). The stringency of the threshold chosen determines the amount of space sampled. For example, a small allowable deviation such as 1% will result in nearly complete sampling of the space, while a more lax criterion such as allowing up to 50% deviation before further refinement was pursued would result in almost none of the space being sampled. We have found that using AMR with a well chosen refinement threshold can result in a 100 fold reduction of resources expended without a corresponding reduction in the information value of the data gathered from the model parameter space, allowing for a rapid exploration of parameter spaces, thereby dramatically shortening the cognitive model revision cycle (Best et al. 2008).

AMR techniques, because they attempt to sample minimally, may produce local spikes in the data, especially when applied to stochastic models such as the ACT-R spaces described here (i.e., the means are less stable when using fewer model runs). We have found that the inclusion of smoothing as a post-process for AMR generally produces improved results, especially at lower sampling rates, since it uses information from the local neighborhood to cancel out noise present in the surface. We implemented smoothing, as is commonly done in digital image processing, by combining the AMR determined value of a dependent measure at a point in some proportion (e.g., ½ was useful in many of our experiments) with the average of its nearest neighbors on the AMR surface (Plewa et al. 2005).

As parameter spaces become larger and more complex (i.e., greater dimensionality and finer granularity), however, the required resources can prohibit exploration, even with the gains from AMR. The main reason for this is that the scaling of a parameter space is exponential, and thus even relatively simple models may easily exceed the capacity of available computational resources in a typical lab setting. In this situation, high performance computing (HPC) must be leveraged, in combination with AMR, if a timely exploration is to be performed. HPC computing typically involves a large network or cluster of computers that can perform model runs in parallel, resulting in a faster exploration of complex parameter spaces. This is especially useful in the case of cognitive model explorations, which can be described as "embarrassingly parallel", a term used in the field of computational complexity that means that the processes to be parallelized (individual model runs) do not interact with each other (Dutra et al. 2003).

The remainder of our presentation will focus on applying AMR to a set of task models of increasing complexity, demonstrating the utility of AMR and the value of parameter exploration for understanding cognitive models. The three tasks we will describe are the Paired Associates Task (PAT), taken directly from the ACT-R tutorial units (ACT-R Tutorials, 2009), the Psychomotor Vigilance Test (PVT; Dinges & Powell, 1985), and the Walter Reed Serial

Addition and Subtraction Task (SAST; Thorne et al., 1985). We now turn to these models and an exploration of their parameter spaces using AMR and HPC.

Parameter Space Descriptions

The Paired Associates Task, as described in Anderson (1981) is a learning task that involves presentation of 20 nouns associated with the digits 0-9. The pairs are presented once during a study session and then presented 7 times during a testing session. The participant is scored on latency to correct response and proportion of correct responses out of the 20 pairs for each of the 7 presentations.

This task is used as the target of a modeling unit in the ACT-R tutorials where the focus is on understanding the interactions of parameters related to activation in producing the memory behavior of the ACT-R architecture (and its corresponding explanation of human memory). However, the modeling task itself poses a challenge to the novice cognitive modeler, and prospective modelers may leave the tutorial unit unsure of the interactions of the parameters, and possibly even somewhat frustrated. We thus chose this model as a target to see if the methods we have developed could be quickly applied to aid in understanding the behavior of the architecture and model of this task.

In the ACT-R architecture, the latency of a retrieval from declarative memory is impacted by the activation of chunks, where that activation is a product of its base level activation and a noise factor. The activation is also impacted by the rate of decay in declarative memory, while the ability to retrieve activated chunks is impacted by the retrieval threshold, which determines an activation level below which chunks cannot be retrieved. Of these parameters, the base level learning parameter is typically left at a default value, leaving us three parameters to choose from for this exploration. Their behavior is given by the following equations. The first equation relates the retrieval time to *A*, the activation of a chunk, and *F*, the latency factor, while the second equation relates the probability of recall for a chunk to the retrieval threshold, τ, the activation of the chunk, *A*,and the noise parameter of the system, *s*.

$$Time = Fe^{-A}$$

$$P(retrieval)_{Chunk_i} = \frac{1}{1 + e^{(\tau - A_i)/s}}$$

To allow for easy visualization, we chose to focus on only two of these remaining parameters, fixing the noise parameter *s* at 0.5, and exploring the PAT space by varying the parameters for the retrieval threshold (τ) and the latency factor (*F*), as suggested in the tutorial instructions (ACT-R Tutorials, 2009). We explored levels of τ from -3 to 0 with a step size of 0.25 and levels of *F* from 0 to 0.45 with a step-size of 0.025, resulting in a space with 13 levels of τ, 19 levels of *F*, and a total of 247 parameter combinations.

Our general approach to understanding the efficiency and effectiveness of AMR methods, which we also used below with the PVT and SAST models, is to first collect 100 model runs at each parameter combination, and then divide these into a "train" and "test" portion of the data. The comparison of these two halves provides a baseline estimate of how well the data fit themselves (model stability), which can be expressed as a baseline Root Mean Square Error (RMSE). AMR variants can then be compared against this baseline to see what additional error, if any, they produce.

Our exploration was conducted using software written to run the ACT-R models and collate the results automatically, allowing the experimenter to initialize experimental settings and then leave the software to continue unaided. The resulting data are then imported into R, which we used, or an alternative statistical analysis and visualization package.

Our focus is on AMR methods, but to demonstrate the efficiency gain these methods can produce, we also conducted an exhaustive homogeneous sweep of the parameter space for comparison. Our hypothesis is that the same scientific conclusions would be reached with either method, one using a fraction of the computational resources, and thus one source of evidence for this hypothesis will be in the quality of the conclusions a modeler might come to viewing the different diagrams. For this purpose, we will present an exhaustively sampled space, labeled "fully explored" (figure 1), and a minimally sampled space that uses AMR to the full extent possible to reduce computation, labeled "minimally explored" (figure 2). In addition, we also present a visualization of the results of the smoothing post-process (figure 3).

Figures 1-3 are of the latency for the 8th simulated recall trial during the PAT, labeled "t8lat DV", which we selected for presentation based on the obvious interaction between τ and *F*. The gray spheres represent parameter combinations at which models were run.

These figures show that increasing the latency factor produces a predominantly linear increase in reaction times when the retrieval threshold is less than approximately -2, but that higher values of the retrieval threshold (closer to 0) produce an interaction with the latency factor. In particular, the latency for retrievals decreases at higher values of τ, since more active chunks are retrieved more quickly or, in cases when a failure to retrieve a chunk happens, the recognition that this is the case happens faster.

It is hard to imagine how a novice modeler might come to understand this space by manually entering parameter values and attempting to understand the rows of data that result, and thus for this reason alone we might suppose that the use of these methods is desirable. Further, the qualitative conclusion that can be reached comparing the smoothed AMR results (figure 3) to the exhaustive results (figure 1) is obvious: the smoothed AMR surface contains much of the qualitative detail of the exhaustively sampled surface, but at a fraction of the computational cost, having been produced using only 1% of the runs present in the exhaustive graph.

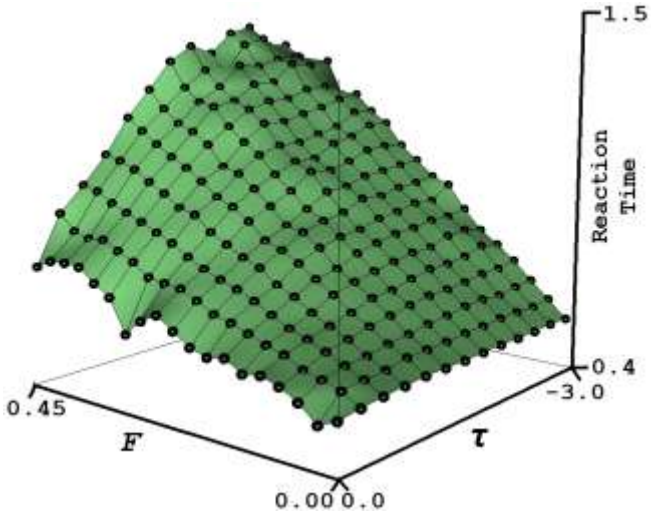

Figure 1: Fully explored parameter space

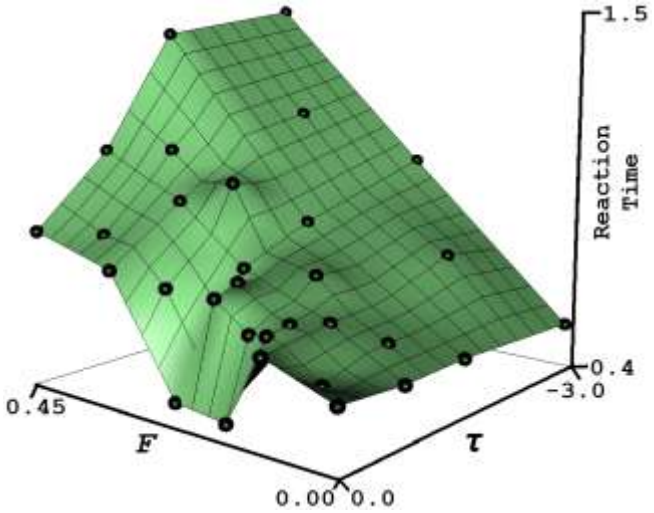

Figure 2: Minimally explored parameter space

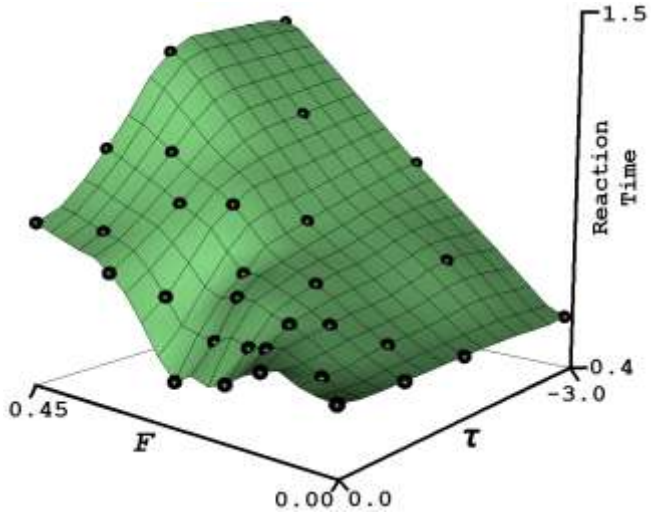

Figure 3: Minimally explored parameter space - smoothed

The question, then, is this: What is the gain of using AMR in terms of computational resources as it relates to any corresponding loss in fidelity? We will now attempt to answer that question both quantitatively and comprehensively in the context of the three tasks we have worked with: the PAT, PVT and SAST. First, however, we will provide a brief background on these two new tasks.

The Psychomotor Vigilance Test involves the presentation of a stimulus at known locations, but at random time intervals, and measuring the time it takes the subject to respond to that stimulus. Responses are binned into 20ms intervals with false starts defined as reaction times faster than 150 ms, lapses as reaction times slower than 500 ms, and sleep attacks as reaction times slower than 30 s. This task, due to its cognitive simplicity and sensitivity to the effects of sleep deprivation and circadian rhythm, is commonly used to assess the impact of fatigue (e.g., Dinges & Powell, 1985; Van Dongen & Dinges, 2005).

The Walter Reed Serial Addition/Subtraction Task involves presenting two single-digit numbers in sequence, followed by an operator – either a plus sign or minus sign. After performing the operation, participants respond with the ones digit of the answer, or the answer plus 10 if the result is negative. Time to correct responses and the percent of correct responses are measured.

As we did with the PAT, these tasks were evaluated within the framework of AMR to determine the impact of AMR methods on accuracy and reduction of computational demands. All of the AMR methods were compared to a corresponding exhaustive parameter sweep, where the exhaustive sweep used 100 model runs at each combination to establish a baseline: the exhaustive data were split in half and compared to determine how well the data fit themselves. This produced a baseline Root Mean Square Error (RMSE) for the model runs against which AMR runs were then compared. In addition, this allowed for an efficiency metric which was simply the percent of the "full space" that was explored by an AMR variant (% Space Sampled). The "full space" is one of the baseline halves and is composed of 50 model runs at each parameter combination. Finally, we also report the total number of model runs involved in each of the spaces and AMR variants. We tested several variations of AMR and smoothing using this methodology. In particular, we examined: 1) allowing the number of model runs to vary as a property of local variation or fixing them at some particular *n*, 2) using local error bounds based on one or all dependent measures, 3) determining local error in dependent measure prediction based on absolute, relative, or statistical criteria, 4) the impact of modifying the smoothing radius and intensity, and 5) the impact of using 4-neighbors vs. 8-neighbors in smoothing. Here we will only report specific instances due to space limitations.

The PVT and SAST spaces have been used to explore the ability of modifications to the ACT-R architecture to account for the pattern of deficits exhibited by people under conditions of extended wakefulness (e.g., Gunzelmann et

al., 2007). These modifications include parameterized mechanisms, which require careful exploration to provide an understanding of their potential impacts. The PVT space was explored using 4 parameters, with the chosen granularity of these parameters resulting in a parameter space with 56,511 parameter combinations. Models were run at each combination for the exhaustive parameter sweep. Similarly, the SAST space was explored by varying 7 parameters, with a necessarily coarser granularity (to partially offset the higher dimensionality) that resulted in a parameter space with a total of 129,600 parameter combinations. Models were run at each of these combinations for the exhaustive parameter sweep.

Table 1: Algorithm Performance Summary

	Data Set	% Control RMSE	% Space Sampled	Total Model Runs
~ 1% Space Sampled	PAT	209.26%	1.07%	132
	PVT	226.45%	1.32%	37,335
	SAST	533.19%	1.09%	70,479
~ 10% Space Sampled	PAT	113.94%	10.20%	1,260
	PVT	154.81%	8.40%	237,350
	SAST	167.55%	9.72%	630,020
100% Space Sampled	PAT	100.00%	100.00%	12,350
	PVT	100.00%	100.00%	2,825,550
	SAST	100.00%	100.00%	6,480,000

In general, with only 10% of the space sampled, for the worst case additional error was only 67.55% beyond the error in the original data when compared to themselves. The granularity of the sampling, however, did interact, and the SAST model, despite having the largest parameter space, also had the coarsest minimum granularity. That is, the SAST has only 6 levels per IV, so not much processing can be skipped, and skipping removes information. The result of this was that, at very sparse sampling of ~1%, the AMR algorithm never proceeded much beyond the initial AMR corners, producing a very rough approximation for SAST. The PVT space granularity fell in the middle of the PAT and SAST spaces, and allowed for dramatic compression with very little loss of accuracy. In particular, in those spaces the error was approximately only doubled (~200% RMSE) when compared to baseline at a very minimal sampling of approximately 1% of the data sampled. This represents a two order of magnitude gain in time to get an answer that, while approximate, is most likely extremely useful (and might, in the case of faulty models, obviate the need for ever collecting the other 99% of the data).

Taken as a whole, algorithm performance is fairly similar across spaces despite dramatic differences in the size of the model spaces. That is, the SAST space is several orders of magnitude larger than the PAT space, but the error terms are within an order of magnitude.

For all three parameter spaces, we explored the effects of performing the homogeneous sweep with a reduced number of model runs. These data are not presented due to space limitations. In all cases, however, AMR methods provided superior results. For example, running 2 models at each parameter combination results in reducing the space sampled to 4%. AMR methods using only 2 model runs result in less space sampled and are more accurate as well.

We also explored adaptively changing the number of model runs at each parameter combination based on measures of local variation. This method ultimately results in focusing computational resources on portions of the space where the model returns spurious results. Increased model runs in these areas does not result in a superior understanding of the model; AMR methods predict these noisy areas more efficiently through linear interpolation.

Conclusions

In this paper, we have demonstrated the application of AMR to a variety of modeling contexts, showing both the visualizations that can be produced and the gains in computational efficiency achieved through this method. In the case of the PAT, the AMR exploration brought out a nonlinear interaction that would most likely not be obvious from a set of tabled values, and would almost certainly be missed by a novice modeler. However, through applying AMR to this task model, we were quickly able to visualize and understand the underlying model and architecture dynamics as a result of examining the impact of varying the parameters that control the model and architecture. This simply cannot be achieved by examining the fit of a model at a particular point in a parameter space.

The PAT could certainly be approached by hand modifying the models in a desktop environment, as it is during the ACT-R tutorials, or even through an exhaustive iterative sweep of the parameter space, but we make the case here that the AMR methods can produce superior understanding with little to no extra investment in computational resources, and thus they are clearly preferable to the alternatives.

As parameter spaces become larger and more complex (i.e., greater dimensionality and finer granularity), the resources required to enumerate or sample from them can become prohibitive, even with the gains from AMR. The reason for this is that the scaling of a parameter space is exponential, and thus even relatively simple models may easily exceed the capacity of available computational resources in a typical lab setting. It is evident that the number of model runs, as reported in Table 1, is a proxy for time. While Moore's Law was once considered a potential way out of computing bottlenecks – simply waiting for faster processors to arrive could solve some issues – that

simply does not apply to problems that scale exponentially. Further, though processors are increasing in speed, our cognitive architectures, models, and the task domains we are interested in are also increasing in complexity, and these effects largely cancel each other out. Thus, it is necessary both to improve the efficiency of our methods through approaches such as AMR, and also to leverage resources that combine processors, such as High Performance Computing (HPC). HPC typically involves a large network or cluster of computers that can perform model runs in parallel, resulting in a faster exploration of complex parameter spaces. This is especially useful in the case of cognitive model explorations, which can be described as "embarrassingly parallel", a term used in the field of computational complexity that means that the processes to be parallelized (individual model runs) do not interact with each other (Dutra 2003).

Fortuitously, these methods also provide a natural gateway to solving harder computational problems: a problem formulated for AMR solution and visualization in the desktop environment is already formulated for HPC solution and visualization.

The techniques described here demonstrate effective ways for exploring large parameter spaces. Indeed, the work described here could not have been conducted without these techniques. This is not to say, however, that the underlying exponential nature of cognitive modeling problems has been tamed. Rather, the methods here provide a significant amount of leverage to a scientist who has managed to reduce the effectively infinite space of cognitive models to a manageable size.

Acknowledgments

The views expressed in this paper are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government. This research was funded by the Air Force Research Laboratory's Warfighter Readiness Research Division in Mesa, Arizona.

References

ACT-R Tutorials (2009). Retrieved from the ACT-R Web site: http://act-r.psy.cmu.edu/actr6/.

Anderson, J.R. (1981). Interference: The relationship between response latency and response accuracy. *Journal of Experimental Psychology: Human Learning and Memory*, 7, 326-343.

Berger, M., & Oliger, J. (1984). Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics*, 53, 484-512.

Best, B., Fincham, J., Gluck, K., Gunzelmann, G., & Krusmark, M. (2008). Efficient Use of Large-Scale Computational Resources. In J. Hansberger (Ed.), *Proceedings of the Seventeenth Conference on Behavior Representation in Modeling and Simulation* (pp. 180-181). Orlando, FL: Simulation Interoperability Standards Organization.

Brooke, J.M., Marsh, J., Pettifer, S., and Sastry, L.S (2007). The importance of locality in the visualization of large datasets. *Concurrency and computation: practice and experience*, 19:195–205.

Dinges, D. F., & Powell, J. W. (1985). Microcomputer analyses of performance on a portable, simple visual RT task during sustained operations. *Behavior Research Methods, Instruments, & Computers 17(6)*, 652-655.

Dutra, I. C., Page, D. Santos Costa, V. Shavlik, J.W. and Waddell, M. (2003). Towards automatic management of embarrassingly parallel applications. In *Proceedings of Europar 2003*, Lecture Notes in Computer Science. Klagenfurt, Austria: Springer Verlag.

Gluck, K., Scheutz, M., Gunzelmann, G., Harris, J., & Kershner, J. (2007). Combinatorics meets processing power: Large-scale computational resources for BRIMS. *Proceedings of the Sixteenth Conference on Behavior Representation in Modeling and Simulation* (pp. 73-83). Orlando, FL: Simulation Interoperability Standards Organization.

Gunzelmann, G., & Gluck, K. A. (2008). Approaches to modeling the effects of fatigue on cognitive performance. In J. Hansberger (Ed.), *Proceedings of the Seventeenth Conference on Behavior Representation in Modeling and Simulation* (pp. 136-145). Orlando, FL: Simulation Interoperability Standards Organization

Gunzelmann, G., Gluck, K. A., Kershner, J., Van Dongen, H. P. A., & Dinges, D. F. (2007). Understanding decrements in knowledge access resulting from increased fatigue. In D. S. McNamara and J. G. Trafton (Eds.), *Proceedings of the Twenty-Ninth Annual Meeting of the Cognitive Science Society* (pp. 329-334). Austin, TX: Cognitive Science Society.

Gunzelmann, G., Gross, J. B., Gluck, K. A., & Dinges, D. F. (in press). Sleep deprivation and sustained attention performance: Integrating mathematical and cognitive modeling. *Cognitive Science*.

Plewa, T., Linde, T.J., and Weirs, V.G. (2005). *Adaptive mesh refinement, theory and applications: proceedings of the Chicago Workshop on Adaptive Mesh Refinement Methods, Sept. 3-5, 2003.* Springer Verlag.

Rai, M. M., & Anderson, D. (1981). Grid evolution in time asymptotic problems. *Journal of Computational Physics*, 43, 327-344.

Roberts, S., & Pashler, H. (2000). How persuasive is a good fit? A comment on theory testing. *Psychological Review*, *107*, 358-367.

Thorne, D. R., Genser, S. G., Sing, G. C., & Hegge, F. W. (1985). The Walter Reed performance assessment battery. *Neurobehaviora. Toxicology and Teratology*, 7, 415-418.

Van Dongen, H. P. A., & Dinges, D. F. (2005). Sleep, Circadian Rhythms, and Psychomotor Vigilance. *Clinical Sports Medicine, 24*, 237-249.

Willett, R. (2003, September 24). *Sampling Theory and Spline Interpolation*. Retrieved from the Connexions Web site: http://cnx.org/content/m11126/2.3/