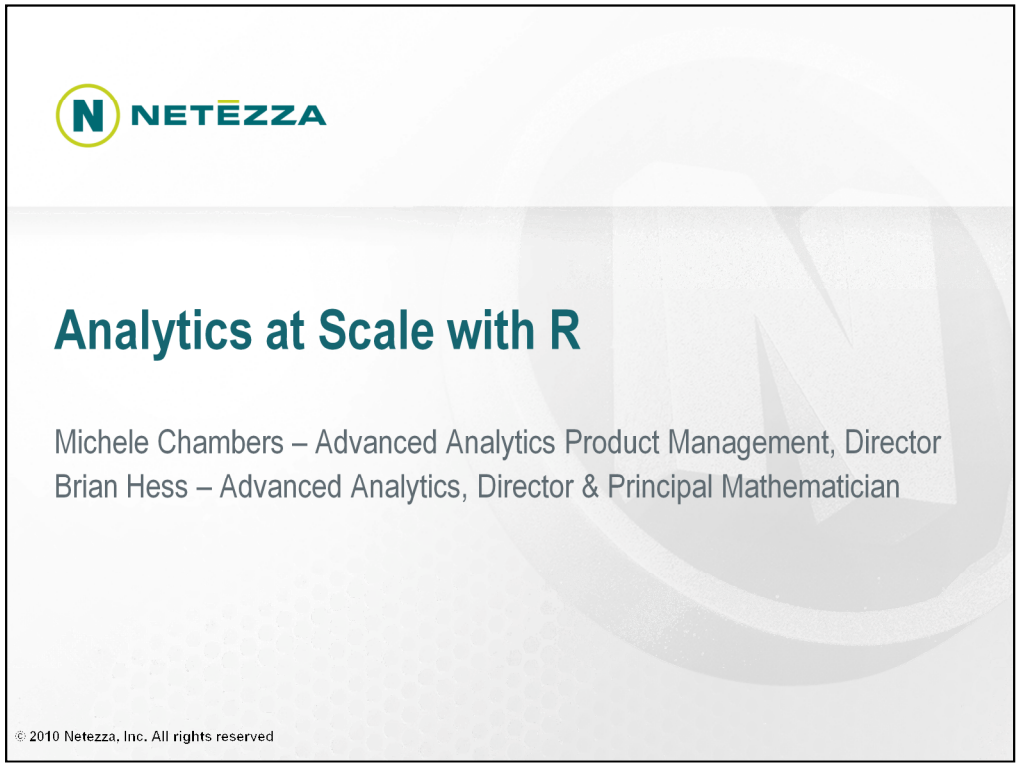


As you may know, Netezza is new to the R world and community. Our focus historically has been large data warehouses.

We are learning through our interactions with customer, prospect and partners that have used or would like to use R.

This chat is a summary of our observations that we'd like to use to start a conversation with all of you on running R at scale. We want our thoughts, observations and experiences to help shape our thoughts and development around R. So, please interject with your thoughts as I go along.

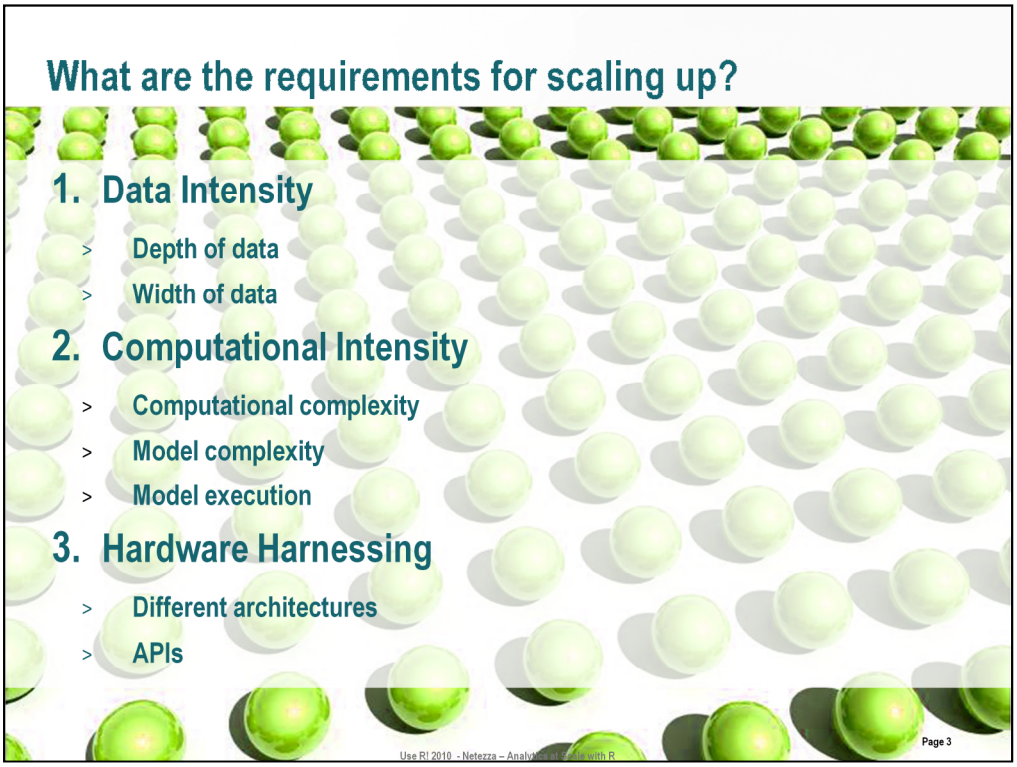
Commercial Acceptance for Enterprise Readiness requires:

- Easy migration from ad-hoc analysis into production deployment
- Ability to learn a model on larger samples and apply the model to large scale data
- Support for real time or near real-time analytics. Which means implies supporting data in motion which means applying a model to streaming data instead of static. (not applicable today for Netezza but what the market is looking for)

Academic or lab environments are on the leading edge addressing large, hard research questions such as:

- Genomic sequencing, microarray analytics, and sensor array projects such as smart grids

Do you have any other use cases for running R at scale?



Data Intensity

- Depth of data (ie: number of transactions, deeper level of transactions or more history) more sources or finer granularity or more regions
- Width of data (ie: number of factors, dimensions, features) more measurements, more observations

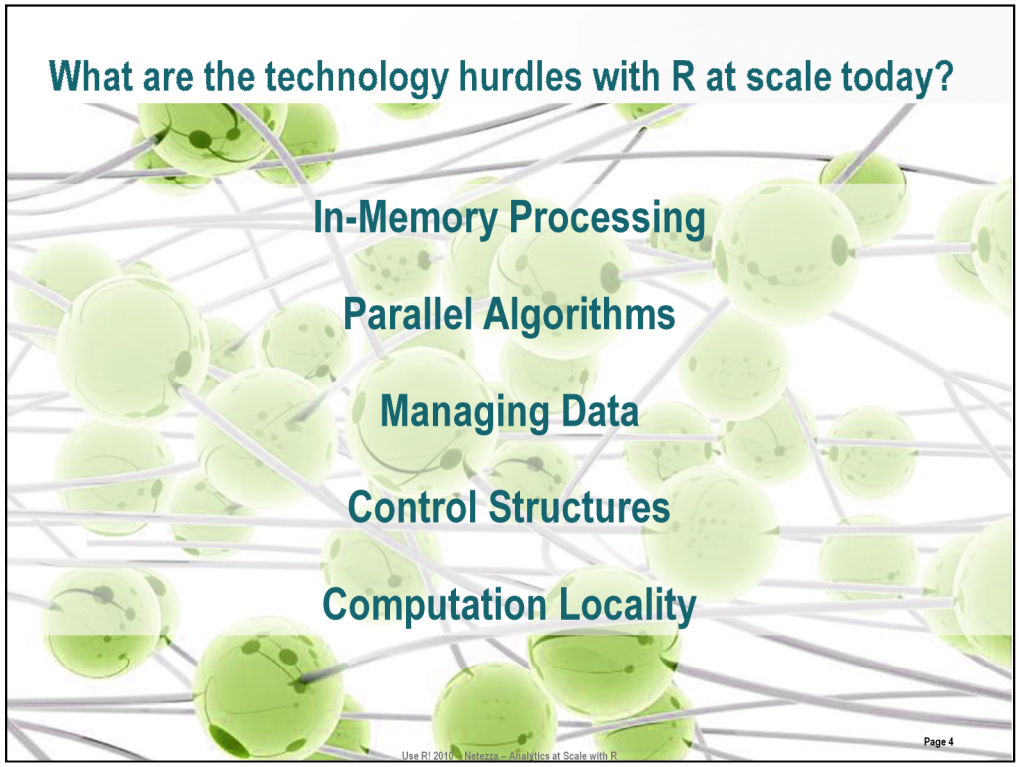
Computational Intensity

- Computational complexity (ie: k-means, PCA, heroic computations, matrix, linear algebra, neural net, the way to combine may be intense)
- Model complexity (ie: number of experiments, simulations, more initial conditions, combination of data mining/prediction/business rules/optimization)
- Model execution (ie: multi-pass operations such as n-fold cross validation, model fitness testing, executing model 1000's times, bootstrapping and combined/hierarchical classification (combining the classifier from a set of simpler ones, e.g. by bagging, boosting), are really computationally intensive and require multiple data passes)

Hardware Harnessing

- Exploit different architectures that are available – new ways to compute that R could benefit from – MPP, grids/multi-node, multi-core, GPU computing (CUDA/OpenCL)
- APIs – Need APIs to be in place to take advantage of underlying hardware – for example some of the work that Revolution Analytics has done – movement in the right direction – set up infrastructure

Any thoughts on what else R would need to have to scale up?



In-Memory Processing

- Limits the amount of data that can be processed
- Data in core can be just small window into full data set
- Penalty for bringing data in/out core piece meal

Parallel Algorithms

- Limited availability of algorithms designed in parallel (SNOW, Sleigh, Revolution Analytics, nzAnalytics)
- Many CRAN packages written in C get data in the form of a double*. That means you can't alter the data representation (like to a distributed RDBMS) because the algorithm was coded to a specific platform/representation. If there was an API in between, then the platform specifics can be put behind the API, and get scalability out of many of the CRAN packages for free.

Managing Data

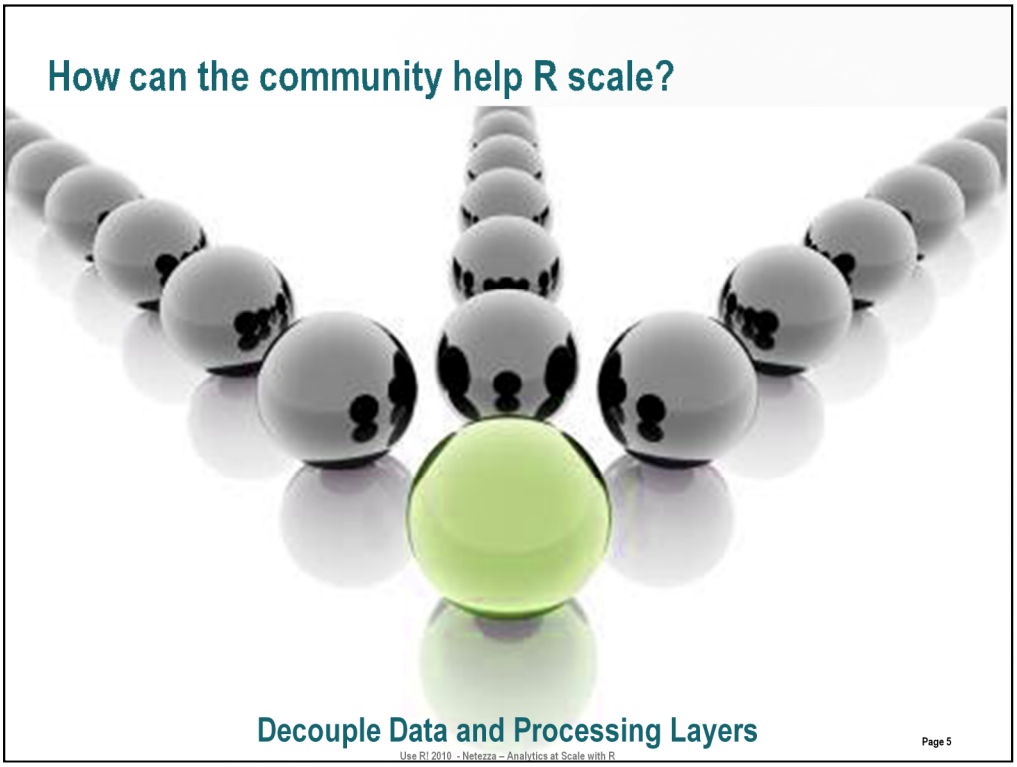
- Data is so large now that if you make mistake you will never finish – so need to be smart so that it's impossible to make mistake – can't brute force your way through problems with large data

Control Structures

- Looping/control structures need to be re-examined to optimize performance
- Don't want to be looping in that you're constantly pulling off disk
- Iterating through data in order – in parallel world, data ordering is expensive – why pay penalty if not needed in order?
- In-memory control is typically ordered – if control structures were allowed to be created then could take advantage of those new control structure without paying penalty for sorting (maybe in time order instead or not sorted at all)

Computation Locality

- Even if R is trying to exploit a parallel back end, have to start with data in memory then distribute out to exploit nodes at backend
- Instead if you move the compute next to data rather than in central/master location you gain scalability
- In distributed environments such as Netezza or Hadoop, data is brought into memory through



Today R has the data and processing layers intertwined. From our observations, we believe that if the data layer in R was abstracted away the analytics processing layer for the algorithm writer then the data layer could more easily be adapted to take advantage of the data storage sources such as databases, Hadoop, grip, SMP

Are there other thoughts on how we, as a community, could help?


NETEZZA
Thank you
Michele Chambers mchambers@netezza.com 508.382.8264
Brian Hess bhess@netezza.com 508.382.8471
Page 6
Use R! 2010 - Netezza – Analytics at Scale with R