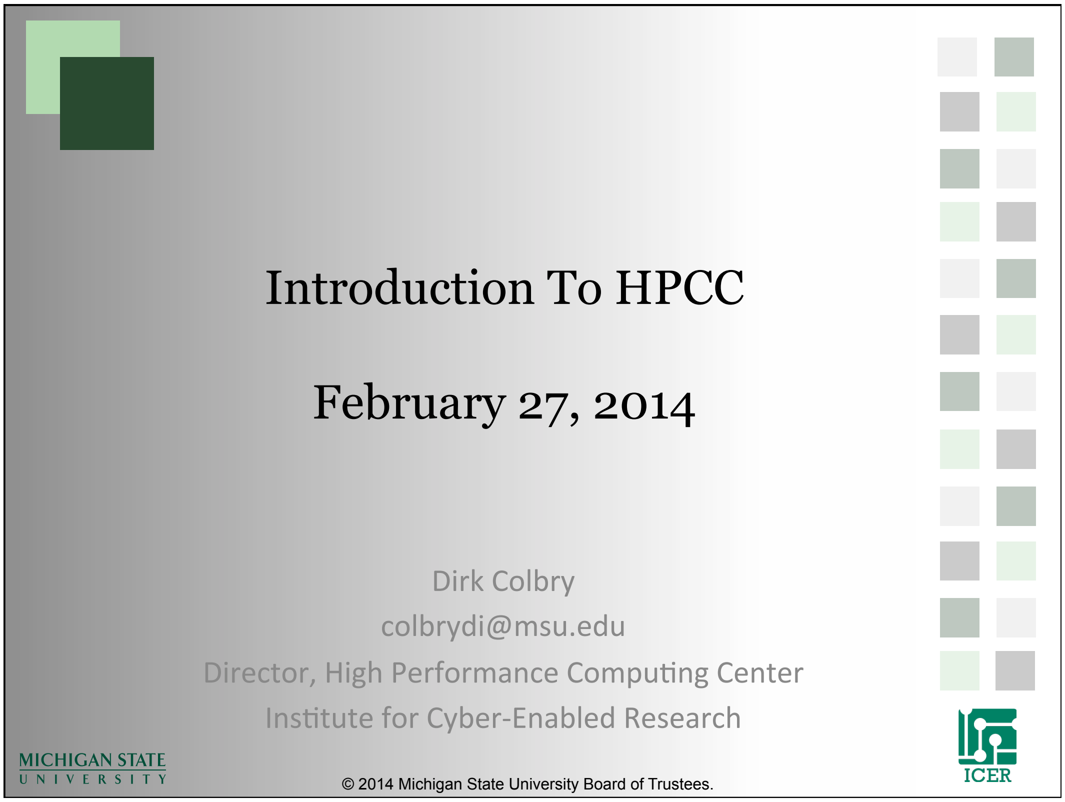


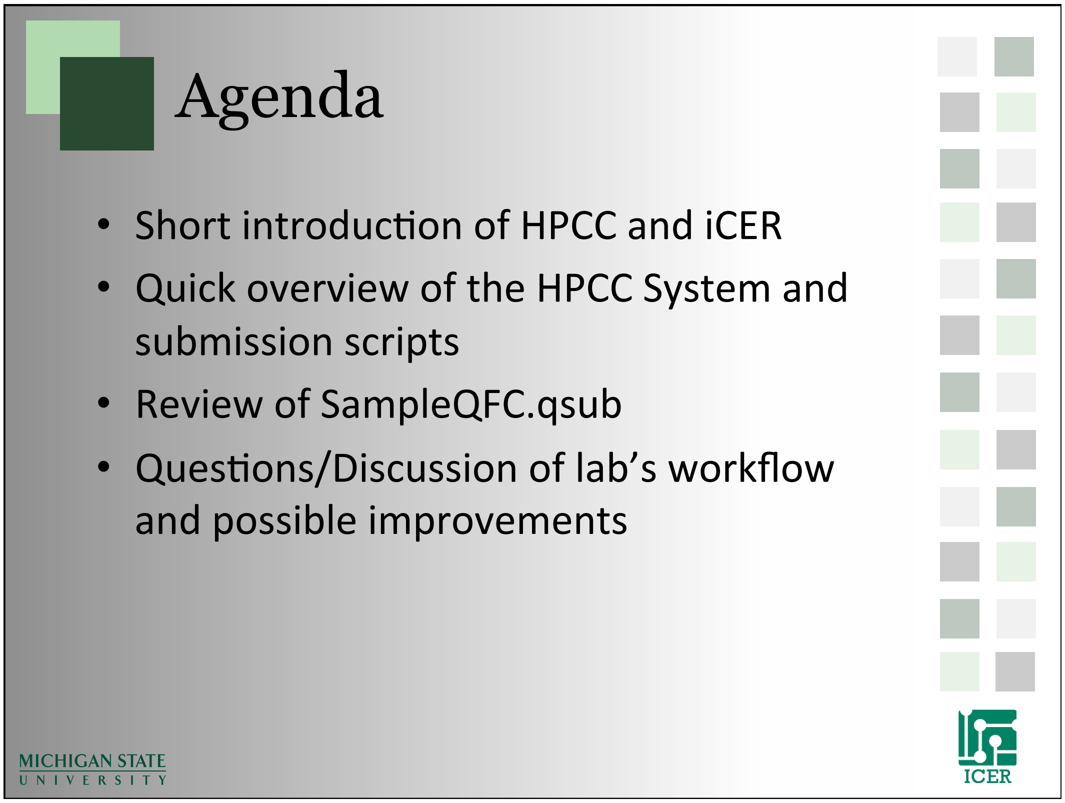
# Introduction To HPCC

February 27, 2014

Dirk Colbry
colbrydi@msu.edu
Director, High Performance Computing Center
Institute for Cyber-Enabled Research

MICHIGAN STATE UNIVERSITY

ICER

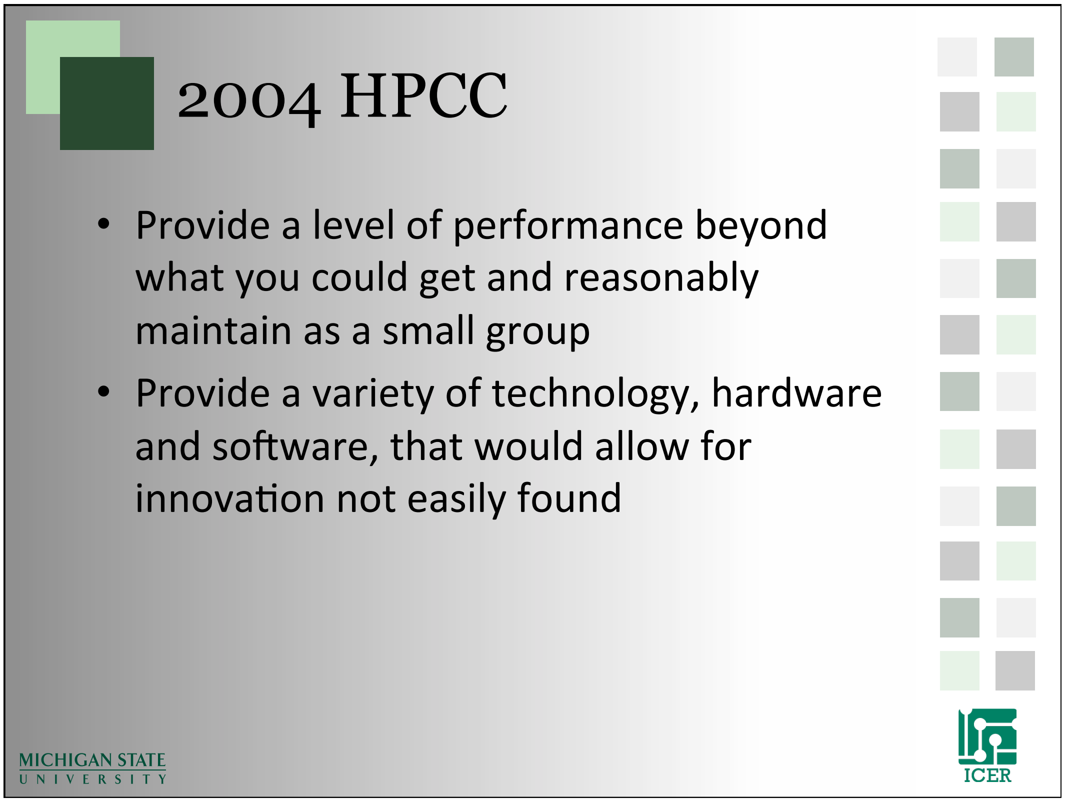


## Agenda

- Short introduction of HPCC and iCER
- Quick overview of the HPCC System and submission scripts
- Review of SampleQFC.qsub
- Questions/Discussion of lab's workflow and possible improvements

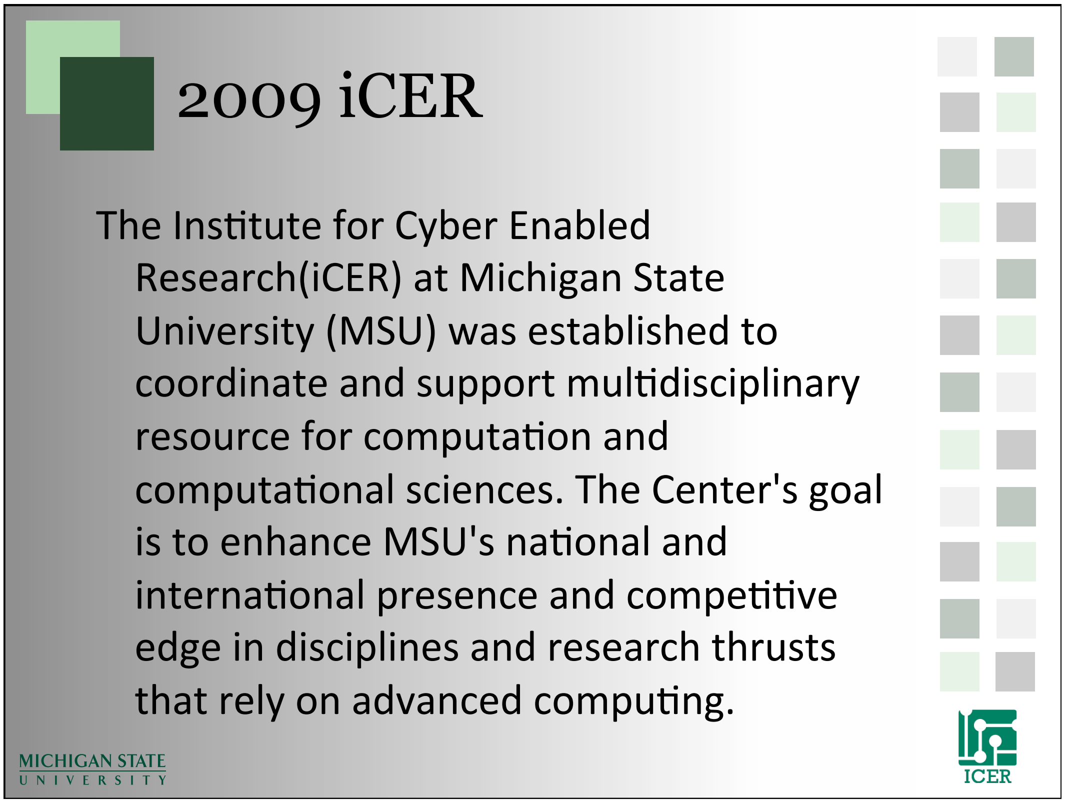
MICHIGAN STATE UNIVERSITY

ICER



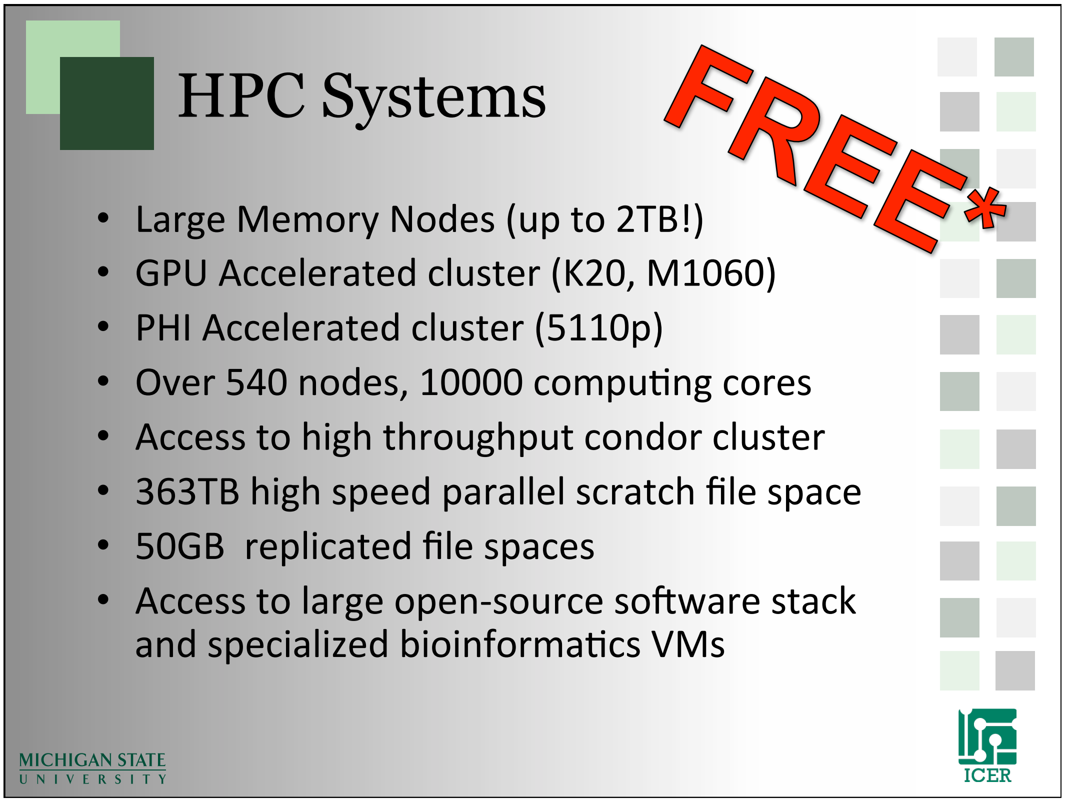
## 2004 HPCC

- Provide a level of performance beyond what you could get and reasonably maintain as a small group
- Provide a variety of technology, hardware and software, that would allow for innovation not easily found



## 2009 iCER

The Institute for Cyber Enabled Research(iCER) at Michigan State University (MSU) was established to coordinate and support multidisciplinary resource for computation and computational sciences. The Center's goal is to enhance MSU's national and international presence and competitive edge in disciplines and research thrusts that rely on advanced computing.

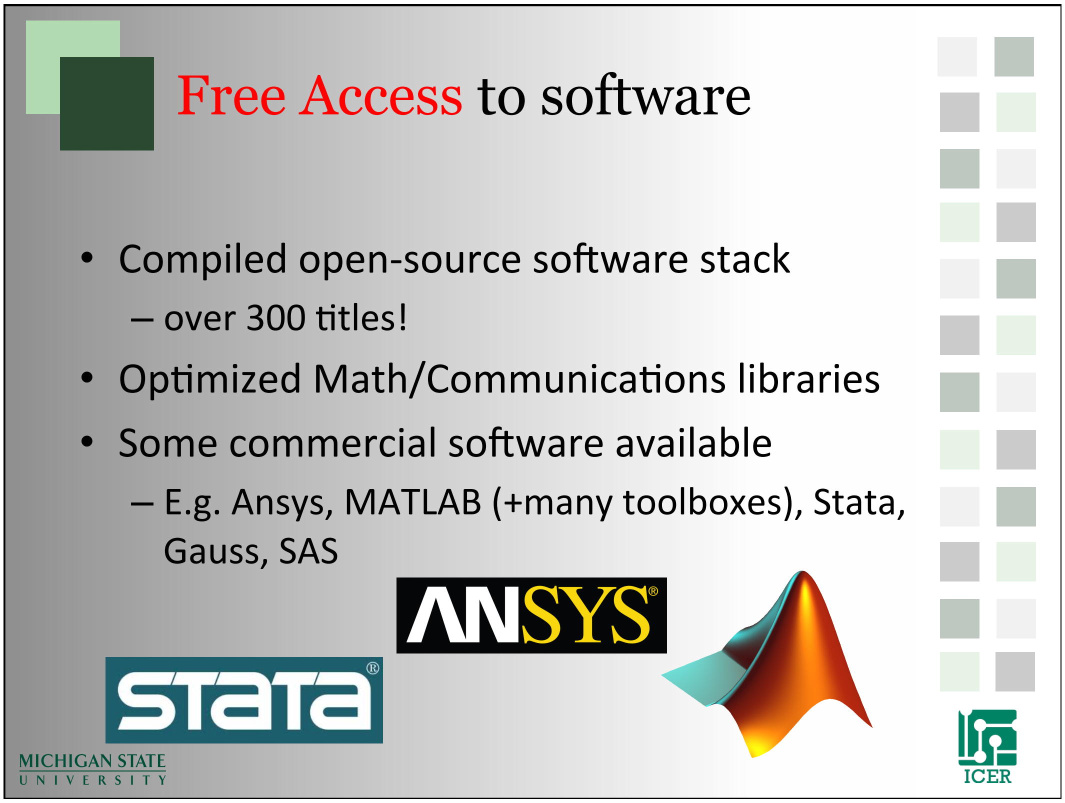


## HPC Systems

FREE*

- Large Memory Nodes (up to 2TB!)
- GPU Accelerated cluster (K20, M1060)
- PHI Accelerated cluster (5110p)
- Over 540 nodes, 10000 computing cores
- Access to high throughput condor cluster
- 363TB high speed parallel scratch file space
- 50GB  replicated file spaces
- Access to large open-source software stack and specialized bioinformatics VMs

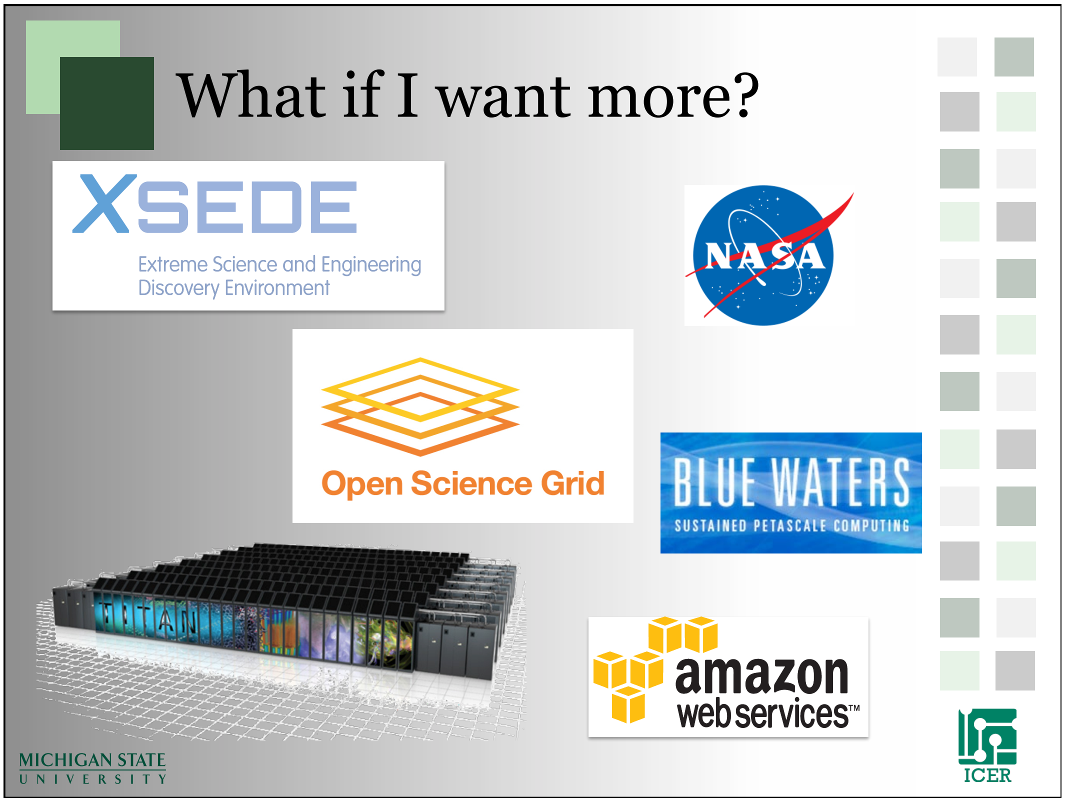
MICHIGAN STATE UNIVERSITY
ICER



## Free Access to software

- Compiled open-source software stack
  - over 300 titles!
- Optimized Math/Communications libraries
- Some commercial software available
  - E.g. Ansys, MATLAB (+many toolboxes), Stata, Gauss, SAS

ANSYS
stata
MICHIGAN STATE UNIVERSITY
ICER



## What if I want more?

XSEDE

Extreme Science and Engineering Discovery Environment

NASA

Open Science Grid

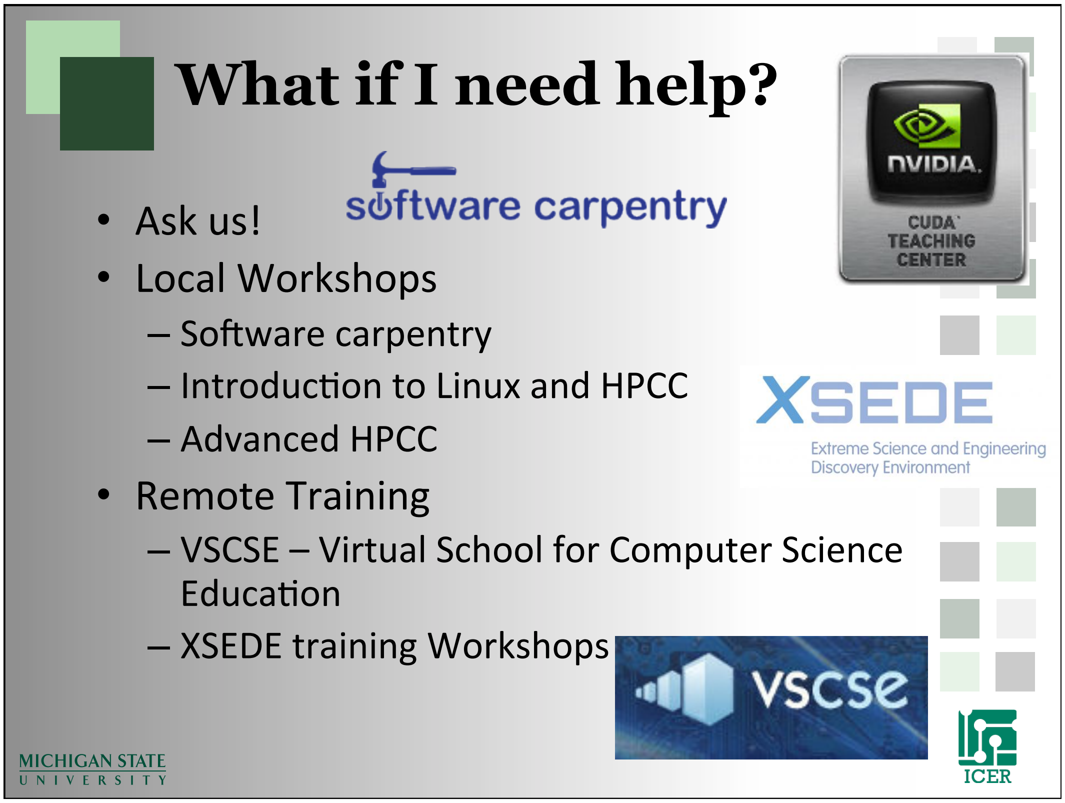
BLUE WATERS

SUSTAINED PETASCALE COMPUTING

amazon web services™

MICHIGAN STATE UNIVERSITY

ICER



## What if I need help?

- Ask us!
- Local Workshops
  - Software carpentry
  - Introduction to Linux and HPCC
  - Advanced HPCC
- Remote Training
  - VSCSE – Virtual School for Computer Science Education
  - XSEDE training Workshops

software carpentry

NVIDIA

CUDA TEACHING CENTER

XSEDE

Extreme Science and Engineering Discovery Environment

vscse

MICHIGAN STATE UNIVERSITY

ICER

# MSU Seminars in Research and Instructional Technology

*May 6,7 2014*

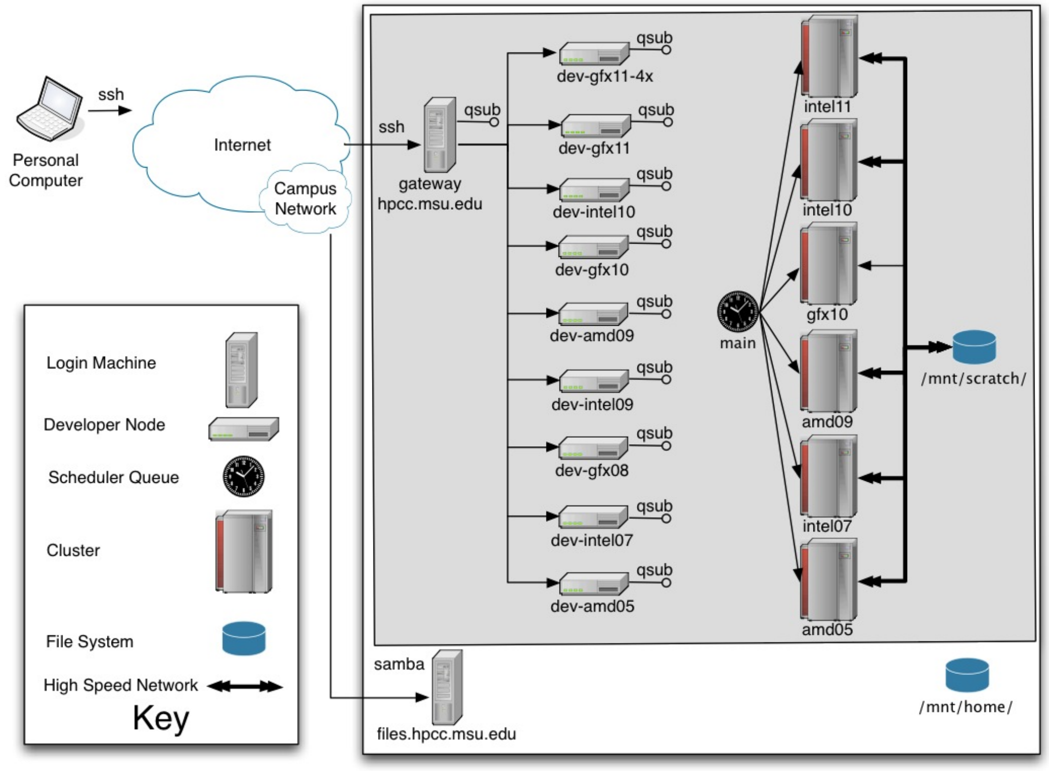
- Two days of no-cost seminars to faculty and graduate students on technology topics.
  - Morning sessions run from 8:30 to 11:30 am.
  - Afternoon sessions run from 1:30 to 4:30 pm.
  - Lunch is provided that will feature guest speakers on instructional technology.

- Tuesday May 6
  - Introduction to HPC
  - Advanced HPC
- Wednesday May 7
  - Managing, Sharing and Moving Big Data

MICHIGAN STATE UNIVERSITY

http://train.msu.edu/faculty/seminars/

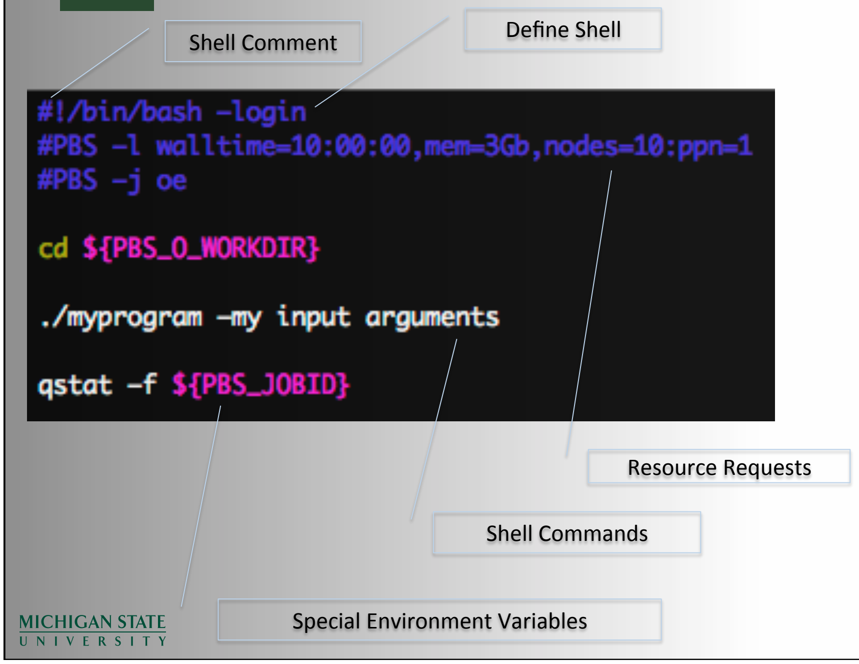
ICER



# Submission Script

1. List of required resources
2. All command line instructions needed to run the computation

# Typical Submission Script



Shell Comment

Define Shell

```
#!/bin/bash -login
#PBS -l walltime=10:00:00,mem=3Gb,nodes=10:ppn=1
#PBS -j oe

cd ${PBS_O_WORKDIR}

./myprogram -my input arguments

qstat -f ${PBS_JOBID}
```

Resource Requests

Shell Commands

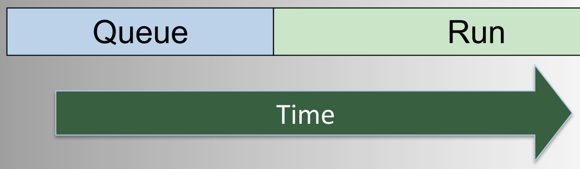
Special Environment Variables

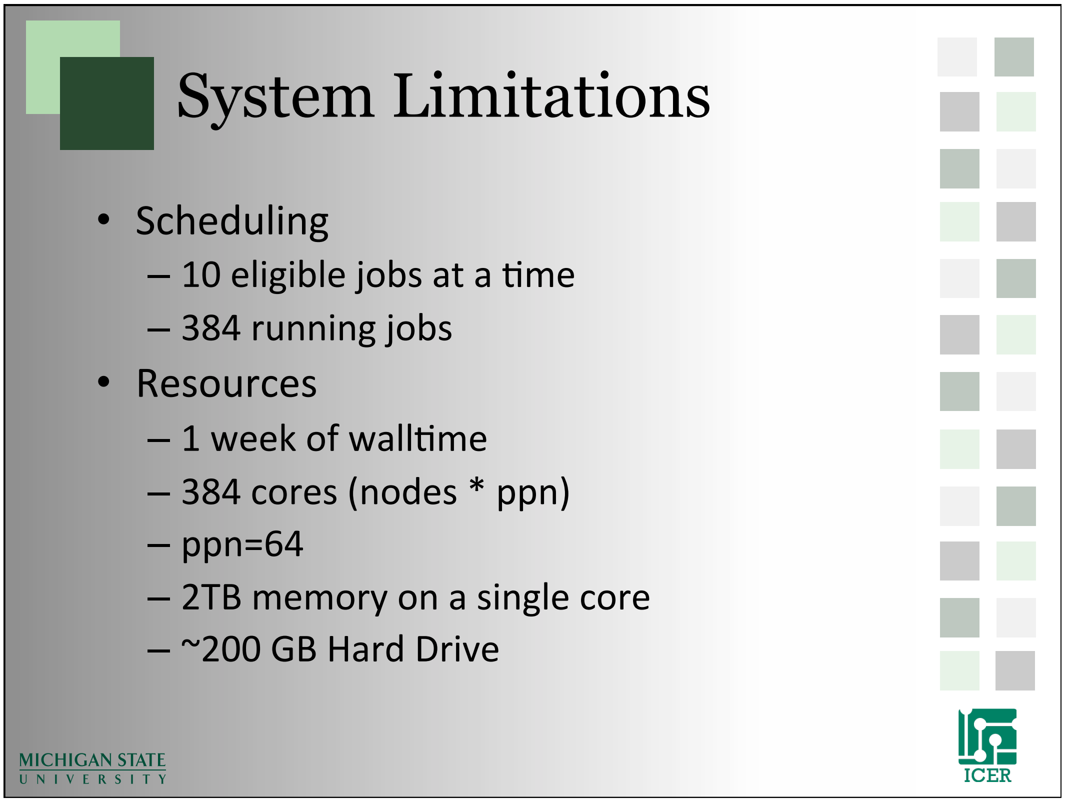
# simple.qsub

```
#!/bin/bash -login
#PBS -l walltime=00:01:00
#PBS -l nodes=1:ppn=1,feature=gbe

cd ${PBS_O_WORKDIR}
./hello

qstat -f ${PBS_JOBID}
```

# Submitting a job

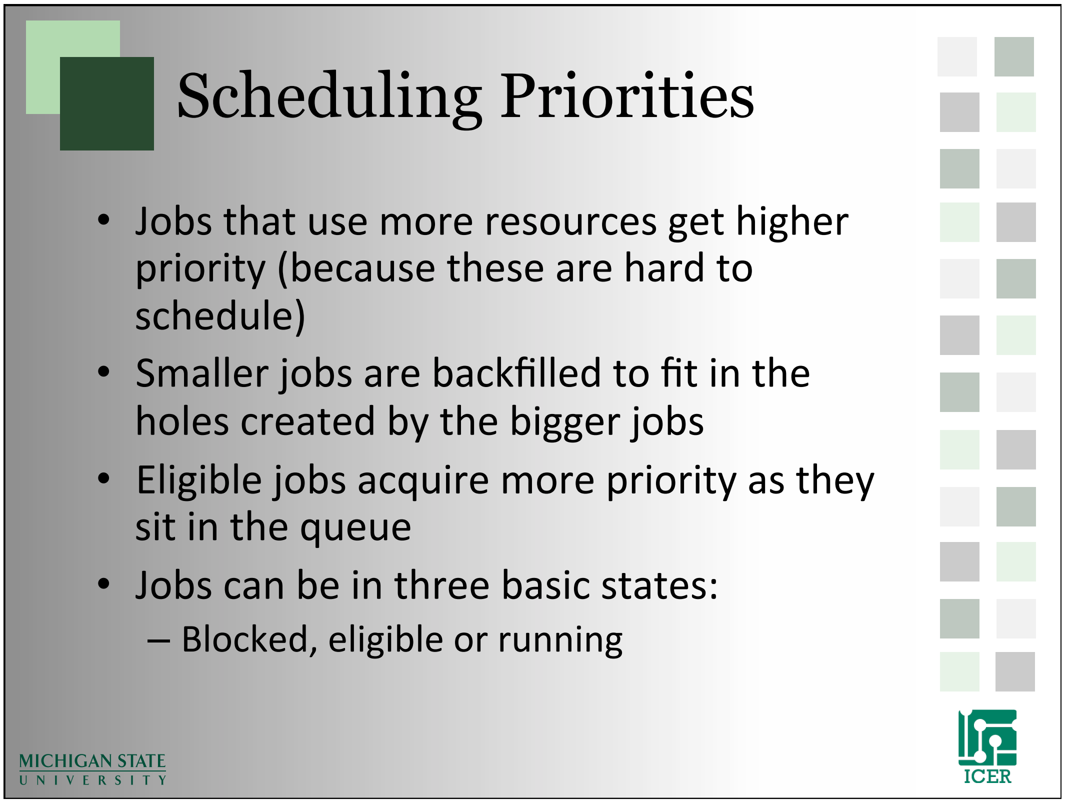
- qsub –arguments <Submission Script>
  - Returns the job ID. Typically looks like the following:
    - 5945571.cmgr01
- Time to job completion





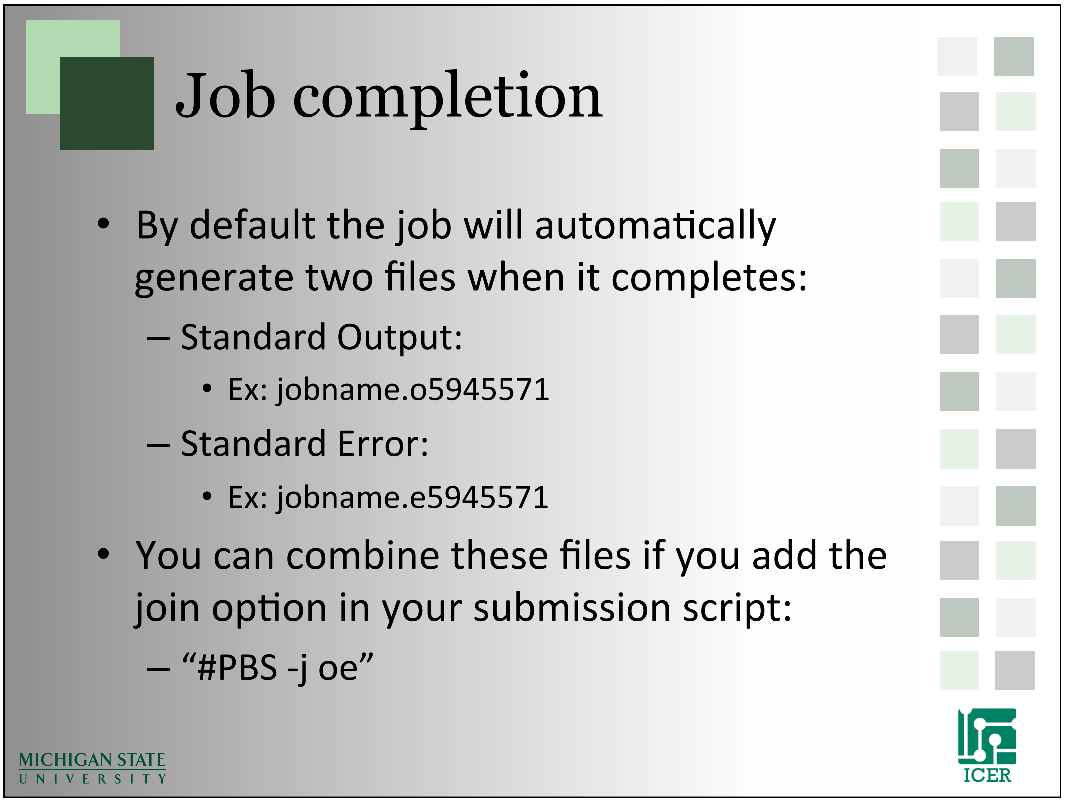
# System Limitations

- Scheduling
  - – 10 eligible jobs at a time
  - – 384 running jobs
- Resources
  - – 1 week of walltime
  - – 384 cores (nodes * ppn)
  - – ppn=64
  - – 2TB memory on a single core
  - – ~200 GB Hard Drive



# Scheduling Priorities

- Jobs that use more resources get higher priority (because these are hard to schedule)
- Smaller jobs are backfilled to fit in the holes created by the bigger jobs
- Eligible jobs acquire more priority as they sit in the queue
- Jobs can be in three basic states:
  - – Blocked, eligible or running



# Job completion

- By default the job will automatically generate two files when it completes:
  - Standard Output:
    - Ex: jobname.o5945571
  - Standard Error:
    - Ex: jobname.e5945571
- You can combine these files if you add the join option in your submission script:
  - "#PBS -j oe"

MICHIGAN STATE UNIVERSITY

ICER

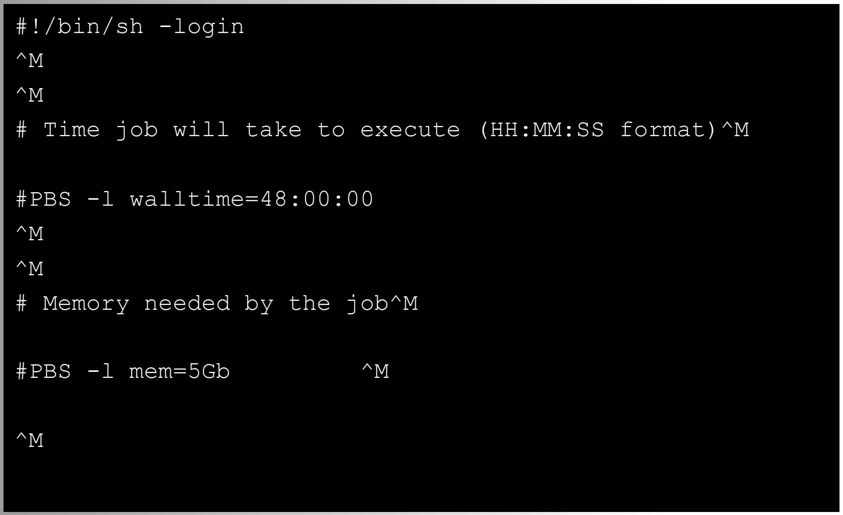
# One-on-one consulting

SampleQFC.qsub

MICHIGAN STATE UNIVERSITY

ICER

# First Look



```
#!/bin/sh -login
^M
^M
# Time job will take to execute (HH:MM:SS format)^M

#PBS -l walltime=48:00:00
^M
^M
# Memory needed by the job^M

#PBS -l mem=5Gb          ^M

^M
```

MICHIGAN STATE UNIVERSITY

ICER

# dos2unix

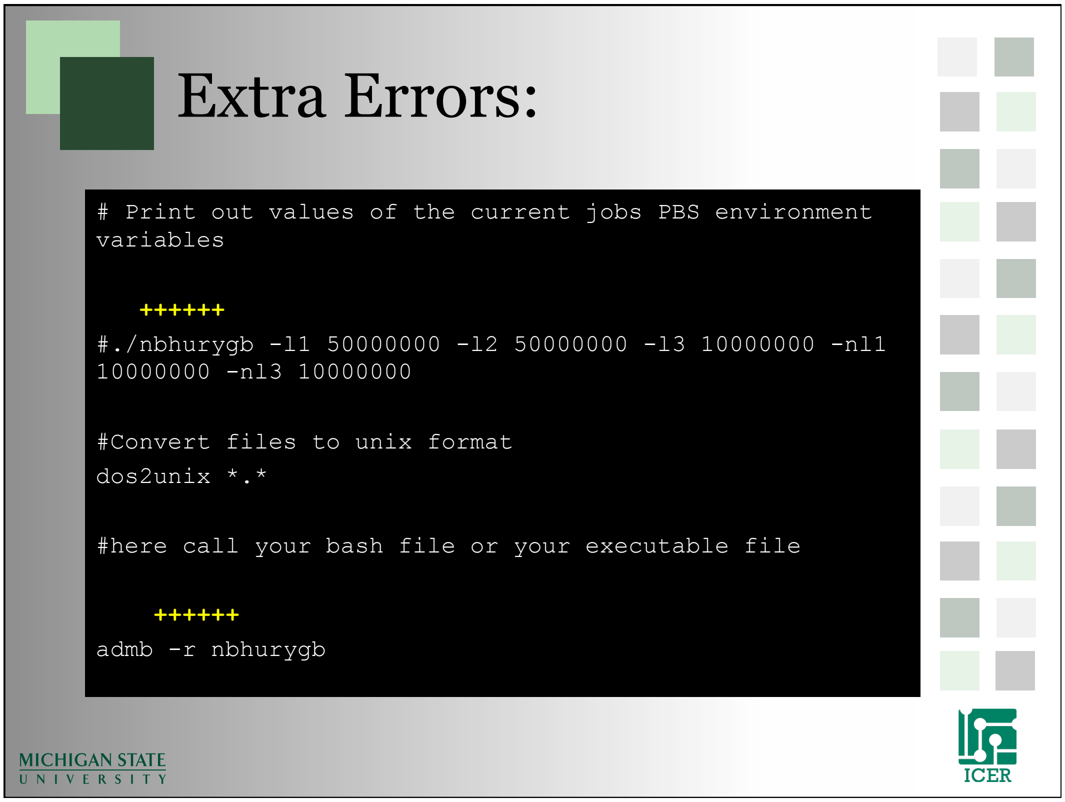
- Converts special end-of-line characters from windows format to dos.

  dos2unix SampleQFC.qsub

- You could also use an editor that uses unix end-of-line characters.
  - I recommend notepad++

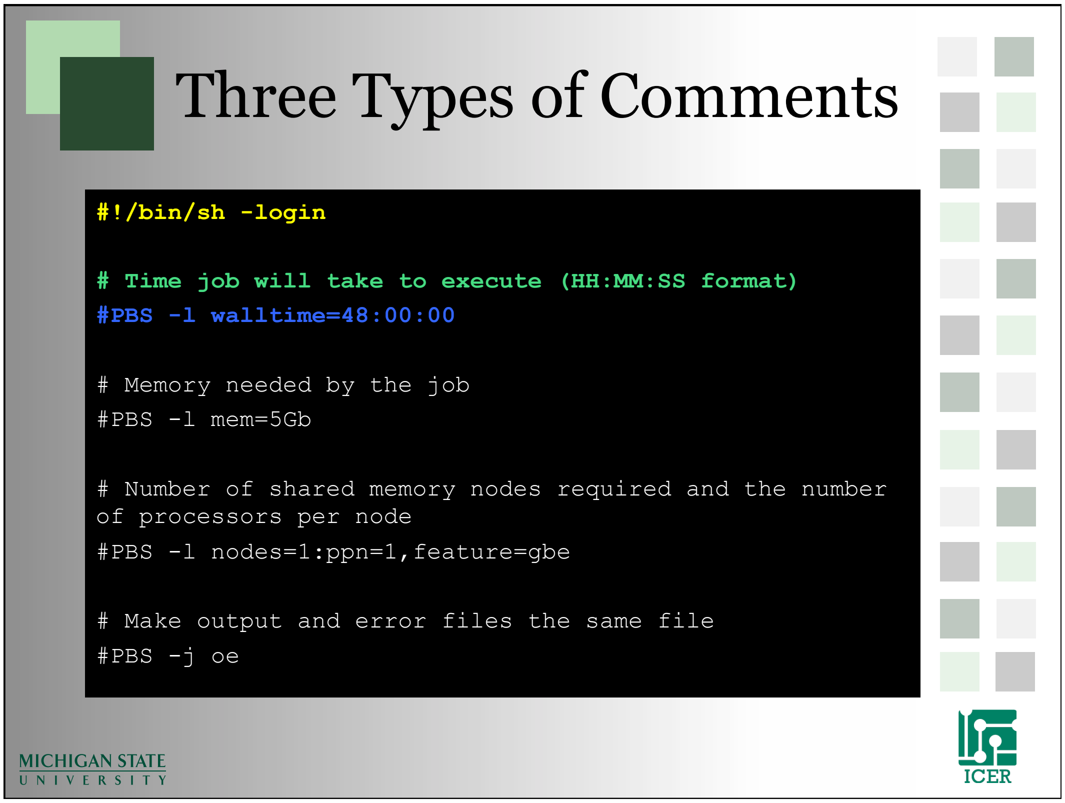
MICHIGAN STATE UNIVERSITY

ICER



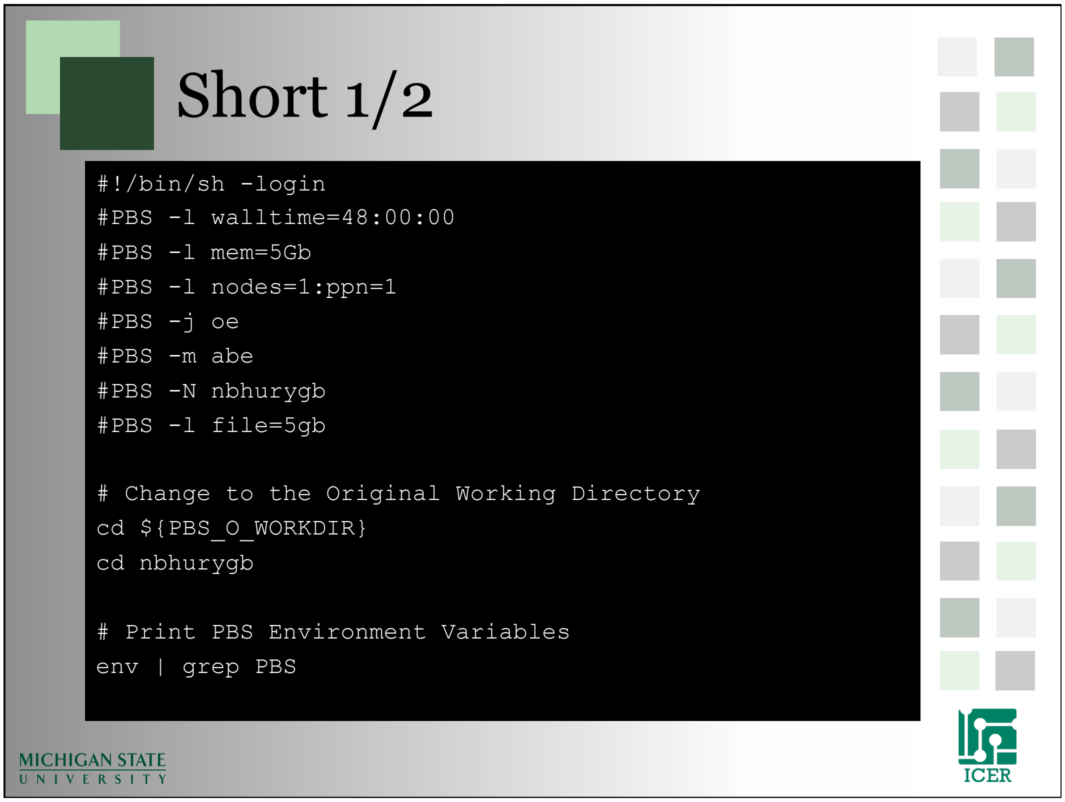
# Extra Errors:

```
# Print out values of the current jobs PBS environment
variables

   ++++++
#./nbhurygb -l1 50000000 -l2 50000000 -l3 10000000 -nl1
10000000 -nl3 10000000

#Convert files to unix format
dos2unix *.*

#here call your bash file or your executable file

    ++++++
admb -r nbhurygb
```



# Three Types of Comments

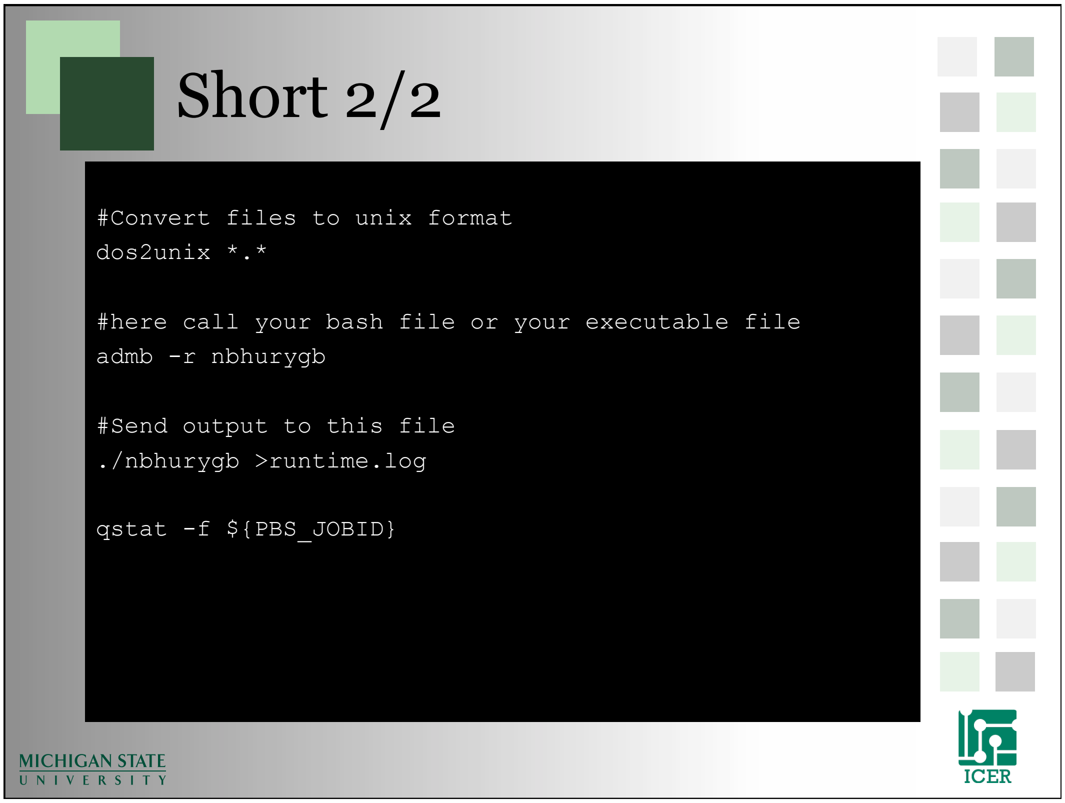
```
#!/bin/sh -login

# Time job will take to execute (HH:MM:SS format)
#PBS -l walltime=48:00:00

# Memory needed by the job
#PBS -l mem=5Gb

# Number of shared memory nodes required and the number
of processors per node
#PBS -l nodes=1:ppn=1,feature=gbe

# Make output and error files the same file
#PBS -j oe
```



## Short 1/2

```
#!/bin/sh -login
#PBS -l walltime=48:00:00
#PBS -l mem=5Gb
#PBS -l nodes=1:ppn=1
#PBS -j oe
#PBS -m abe
#PBS -N nbhurygb
#PBS -l file=5gb

# Change to the Original Working Directory
cd ${PBS_O_WORKDIR}
cd nbhurygb

# Print PBS Environment Variables
env | grep PBS
```

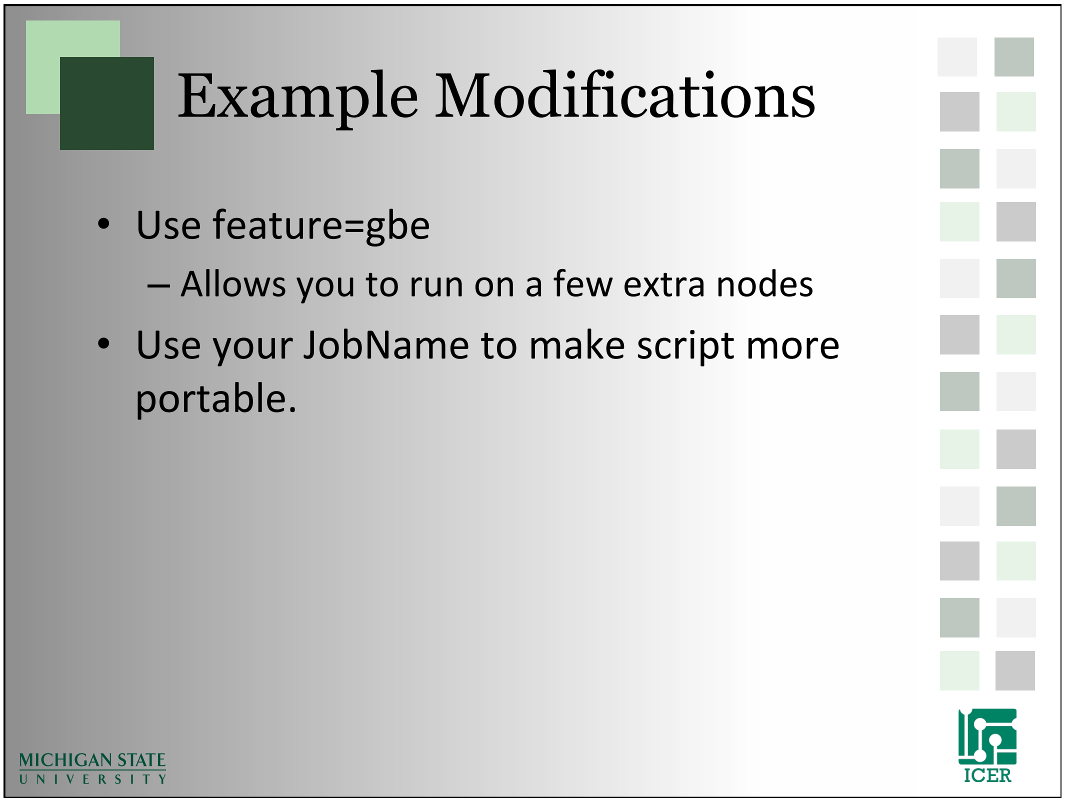
MICHIGAN STATE UNIVERSITY
ICER



## Short 2/2

```
#Convert files to unix format
dos2unix *.*

#here call your bash file or your executable file
admb -r nbhurygb

#Send output to this file
./nbhurygb >runtime.log

qstat -f ${PBS_JOBID}
```

MICHIGAN STATE UNIVERSITY
ICER

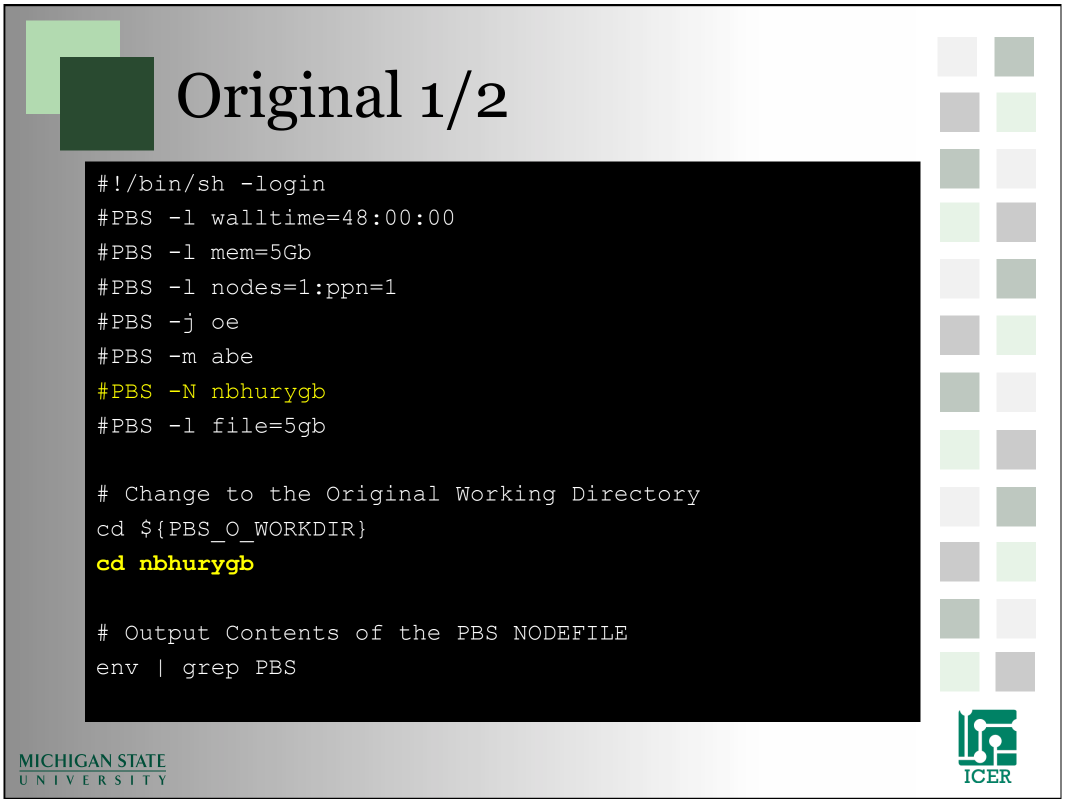


## Example Modifications

- Use feature=gbe
  - Allows you to run on a few extra nodes
- Use your JobName to make script more portable.

MICHIGAN STATE UNIVERSITY

ICER

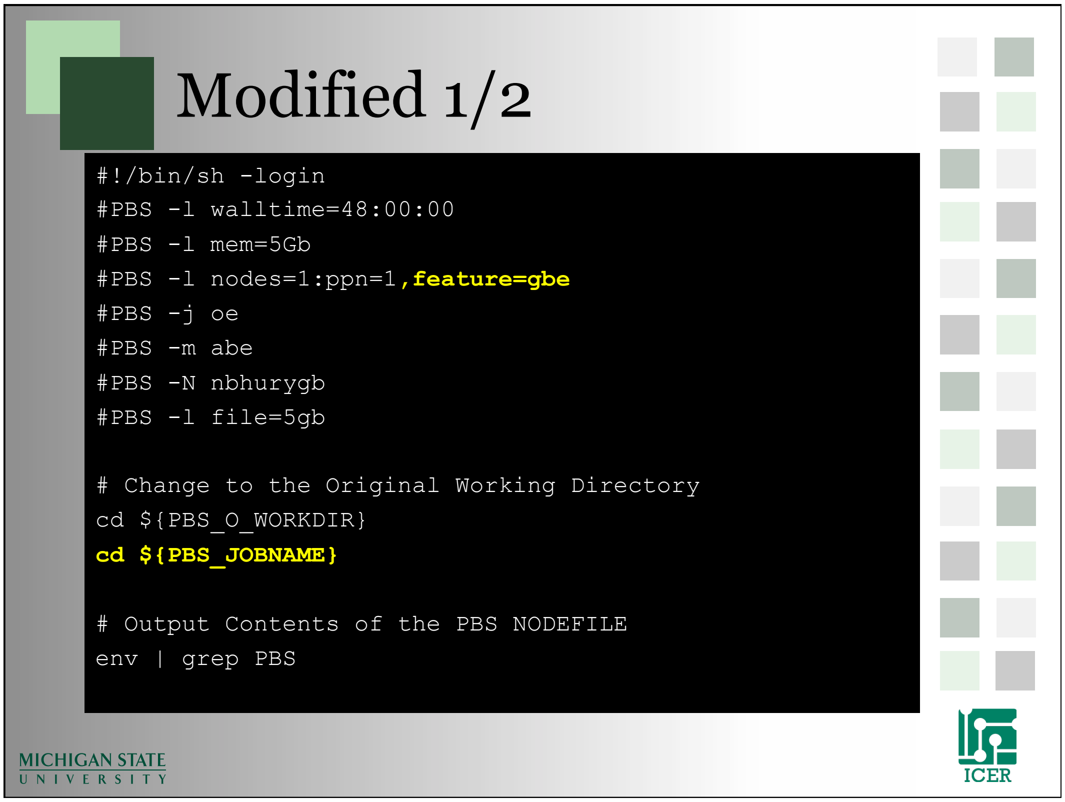


## Original 1/2

```
#!/bin/sh -login
#PBS -l walltime=48:00:00
#PBS -l mem=5Gb
#PBS -l nodes=1:ppn=1
#PBS -j oe
#PBS -m abe
#PBS -N nbhurygb
#PBS -l file=5gb

# Change to the Original Working Directory
cd ${PBS_O_WORKDIR}
cd nbhurygb

# Output Contents of the PBS NODEFILE
env | grep PBS
```

MICHIGAN STATE UNIVERSITY

ICER

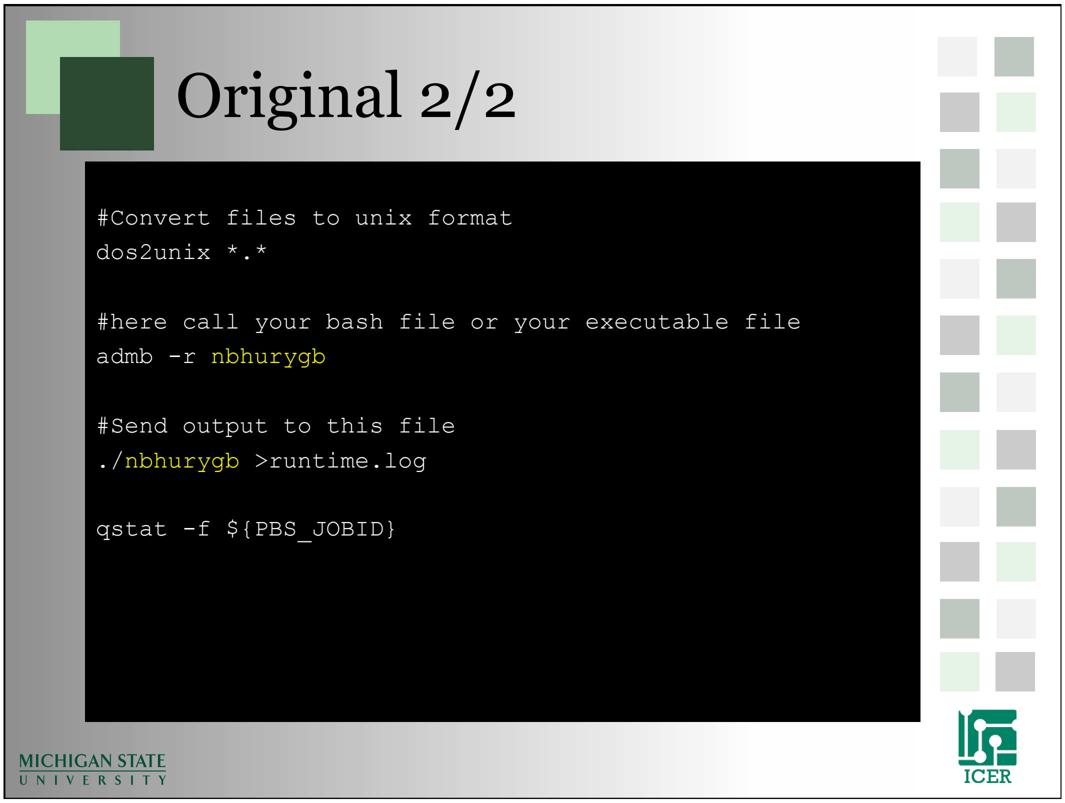


# Modified 1/2

```
#!/bin/sh -login
#PBS -l walltime=48:00:00
#PBS -l mem=5Gb
#PBS -l nodes=1:ppn=1,feature=gbe
#PBS -j oe
#PBS -m abe
#PBS -N nbhurygb
#PBS -l file=5gb

# Change to the Original Working Directory
cd ${PBS_O_WORKDIR}
cd ${PBS_JOBNAME}

# Output Contents of the PBS NODEFILE
env | grep PBS
```

MICHIGAN STATE
UNIVERSITY

ICER

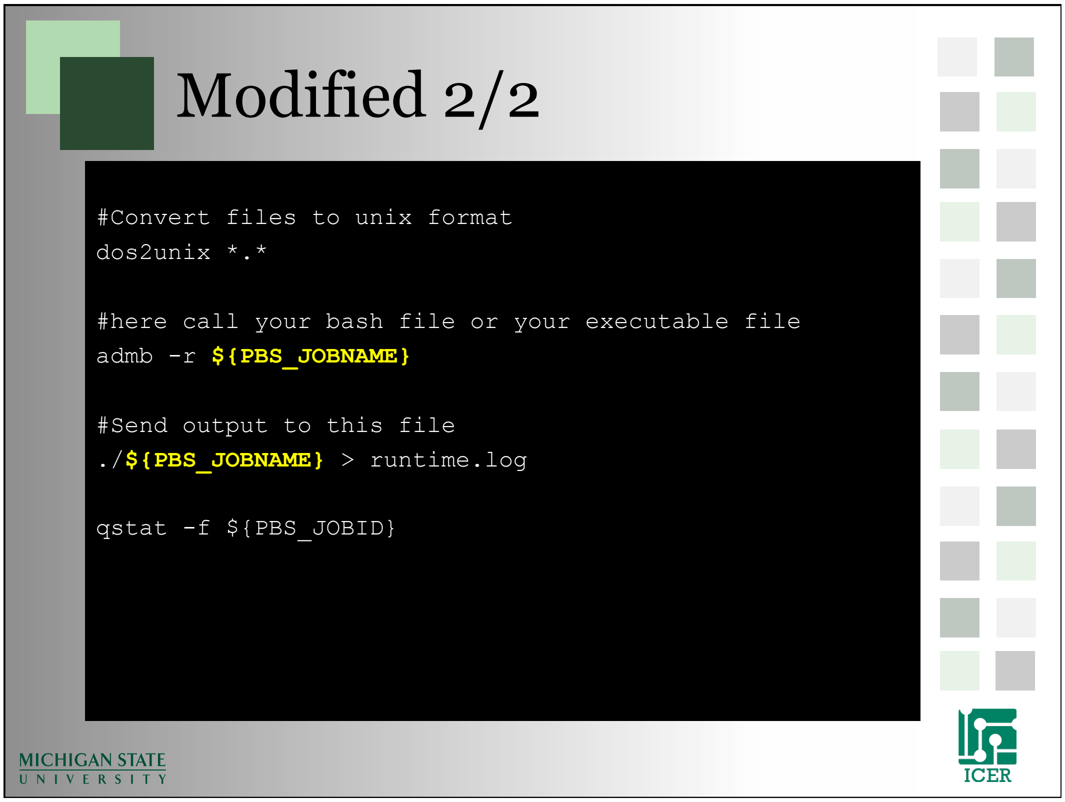


# Original 2/2

```
#Convert files to unix format
dos2unix *.*

#here call your bash file or your executable file
admb -r nbhurygb

#Send output to this file
./nbhurygb >runtime.log

qstat -f ${PBS_JOBID}
```

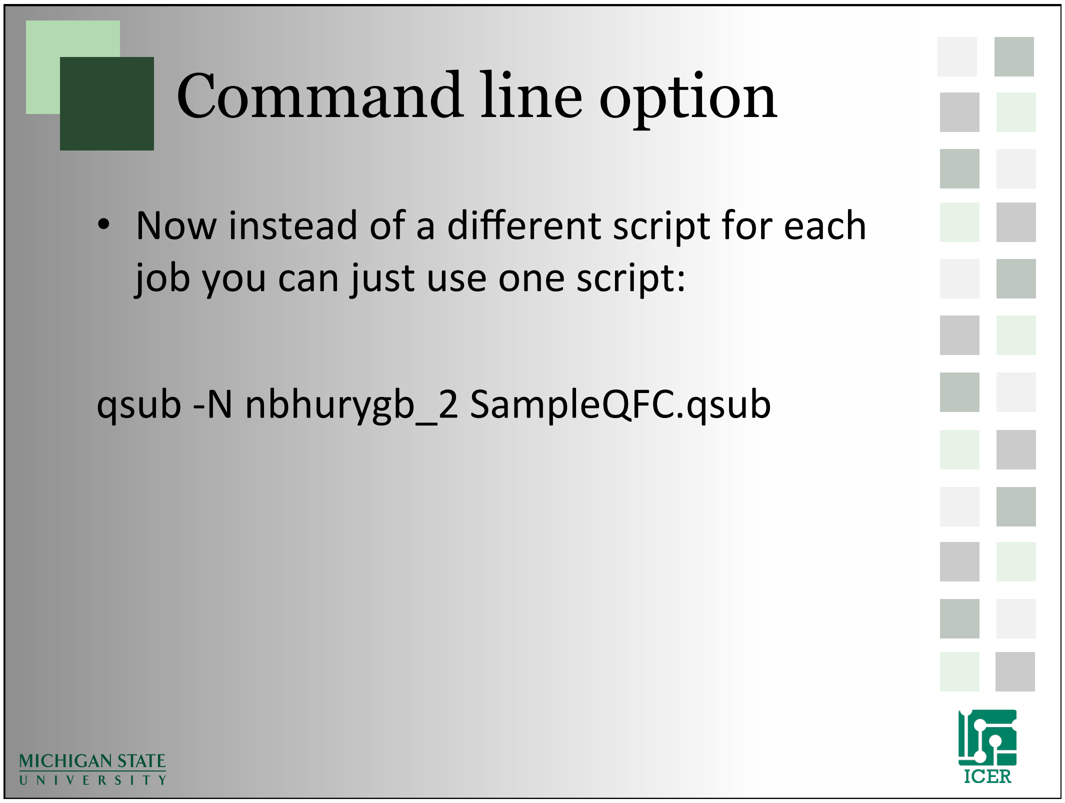
MICHIGAN STATE
UNIVERSITY

ICER



## Modified 2/2

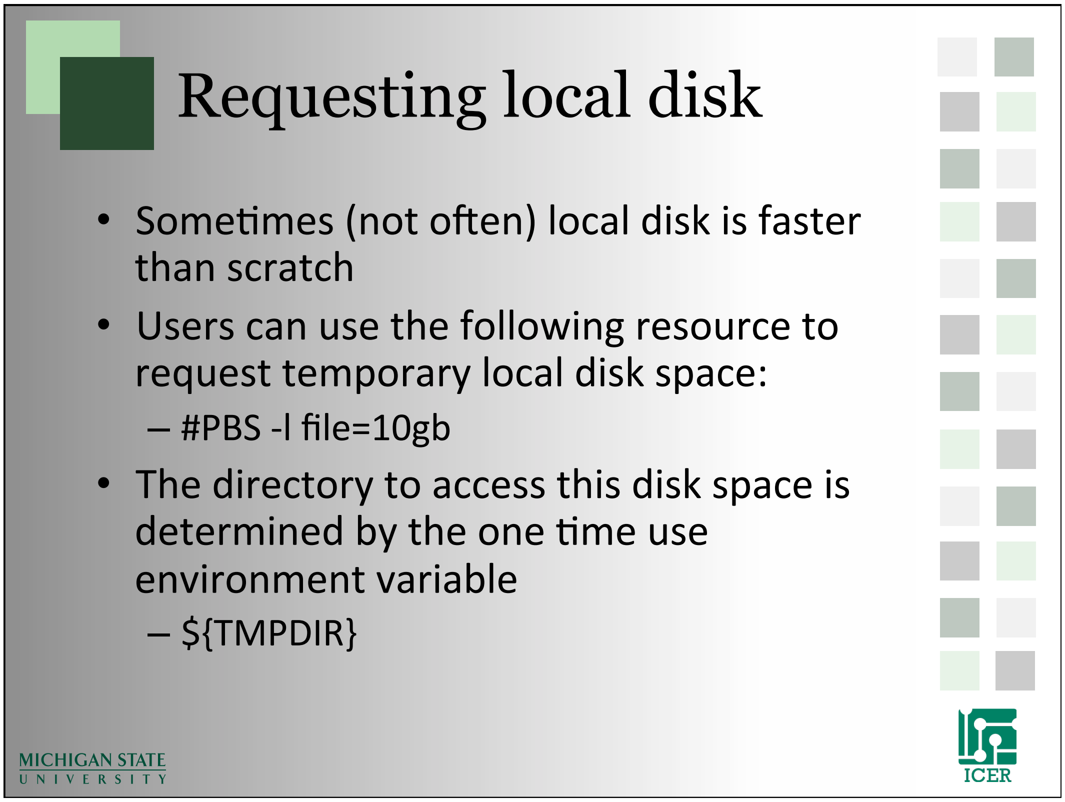
```
#Convert files to unix format
dos2unix *.*

#here call your bash file or your executable file
admb -r ${PBS_JOBNAME}

#Send output to this file
./${PBS_JOBNAME} > runtime.log

qstat -f ${PBS_JOBID}
```



## Command line option

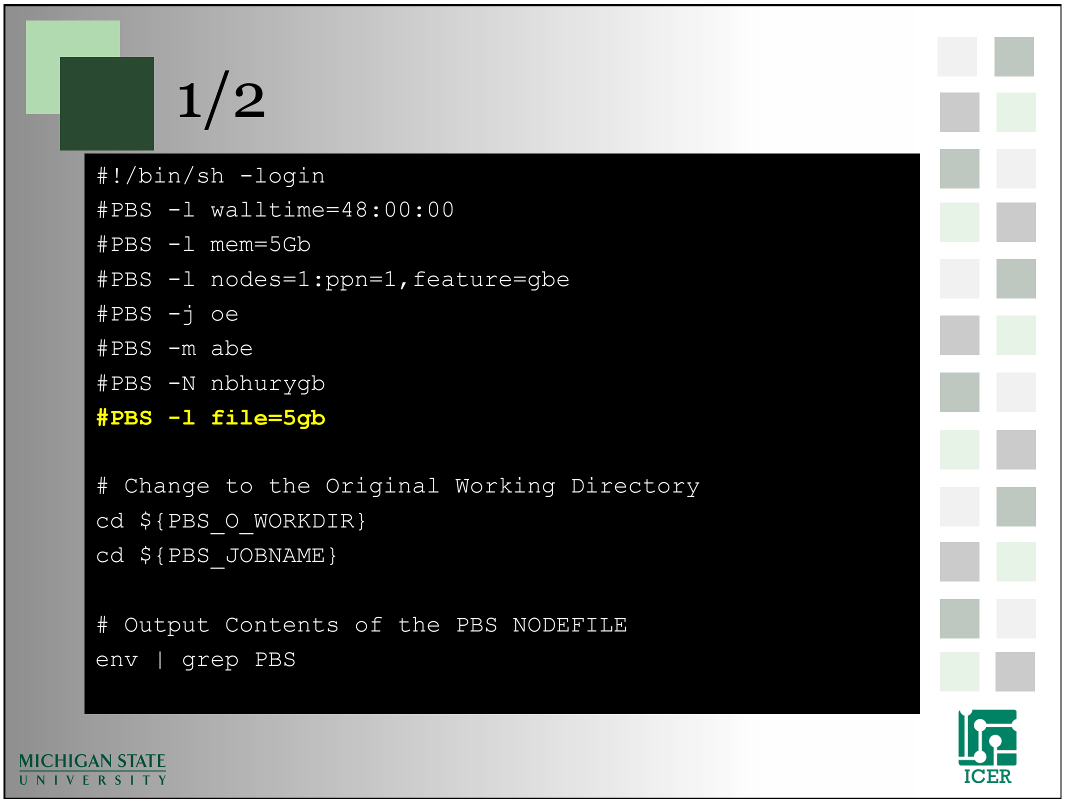
- Now instead of a different script for each job you can just use one script:

qsub -N nbhurygb_2 SampleQFC.qsub



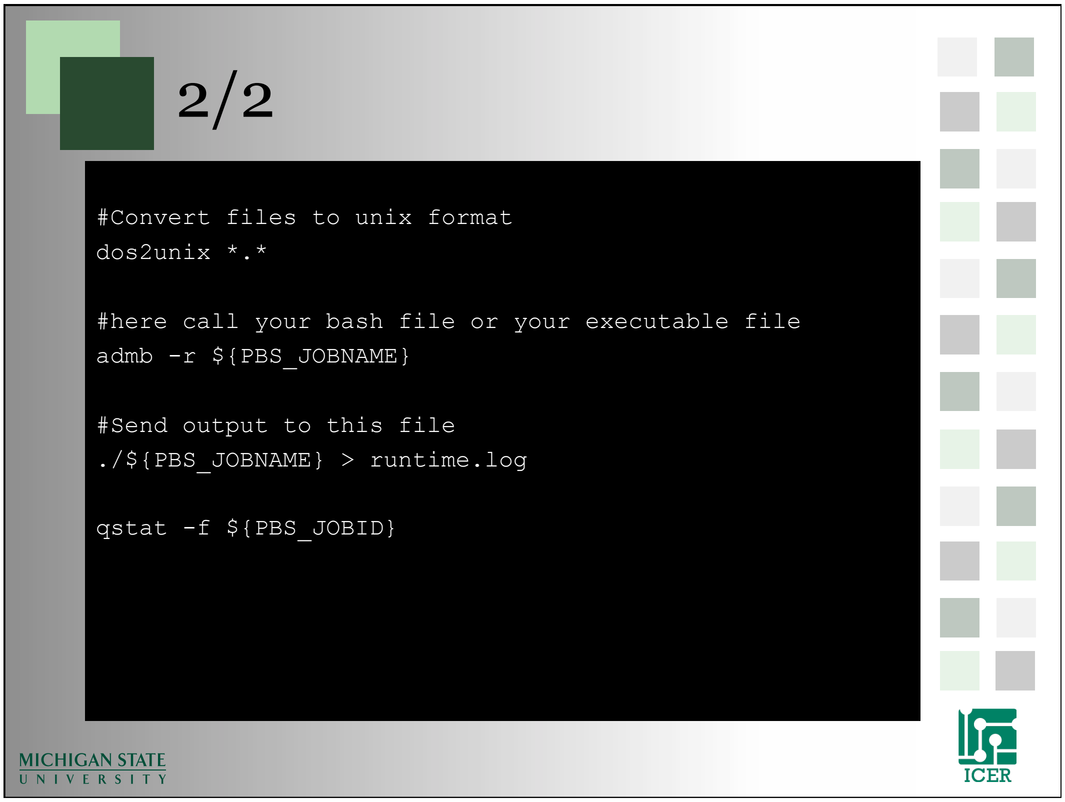
# Requesting local disk

- Sometimes (not often) local disk is faster than scratch
- Users can use the following resource to request temporary local disk space:
  - #PBS -l file=10gb
- The directory to access this disk space is determined by the one time use environment variable
  - ${TMPDIR}



# 1/2

```
#!/bin/sh -login
#PBS -l walltime=48:00:00
#PBS -l mem=5Gb
#PBS -l nodes=1:ppn=1,feature=gbe
#PBS -j oe
#PBS -m abe
#PBS -N nbhurygb
#PBS -l file=5gb

# Change to the Original Working Directory
cd ${PBS_O_WORKDIR}
cd ${PBS_JOBNAME}

# Output Contents of the PBS NODEFILE
env | grep PBS
```



## 2/2

```
#Convert files to unix format
dos2unix *.*

#here call your bash file or your executable file
admb -r ${PBS_JOBNAME}

#Send output to this file
./${PBS_JOBNAME} > runtime.log

qstat -f ${PBS_JOBID}
```

## Discussion