

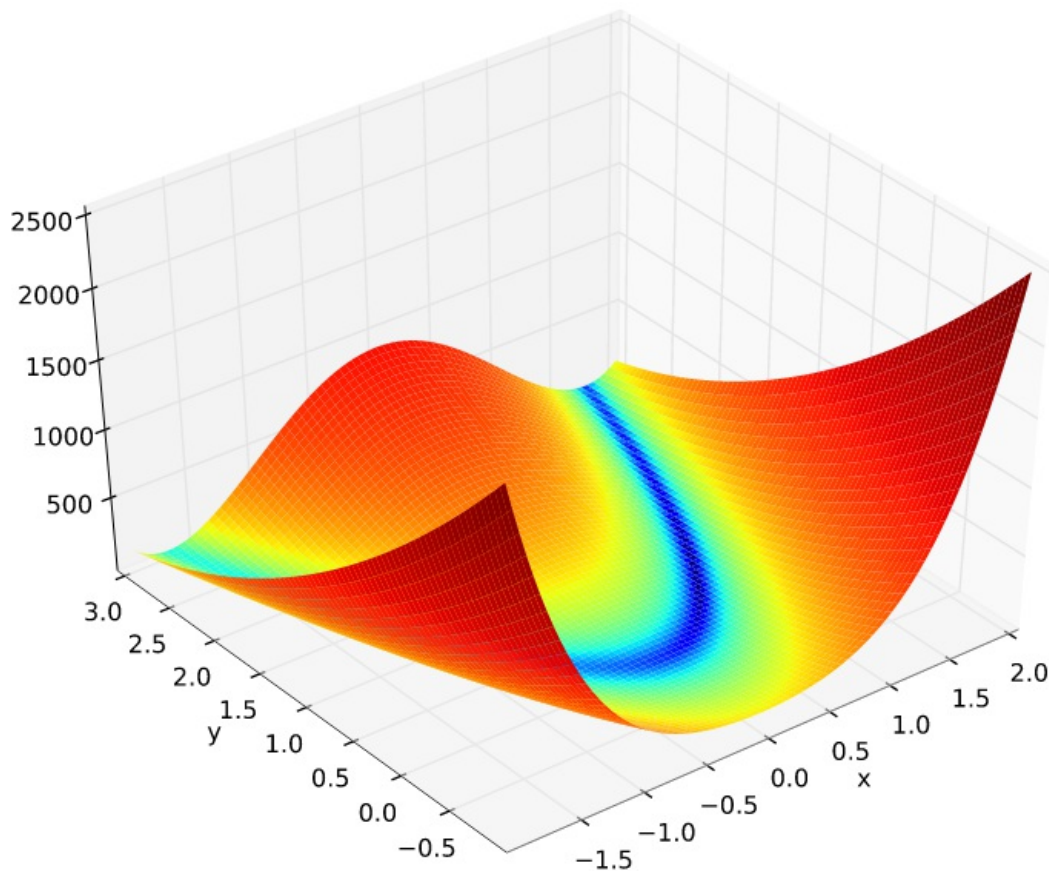
Parametric Sweep with MATLAB and Microsoft HPC Pack

In this example we'll show you how to perform a parametric sweep with MATLAB on a compute cluster of Windows Azure nodes. The basic approach is to compile your MATLAB code (.m files) into a Windows binary executable that accepts the node ID as a command line argument, then use Microsoft HPC Pack to create and launch a parametric sweep job that runs the compiled MATLAB across the cluster.

Our example problem is a simple minimization of the Rosenbrock Banana function:

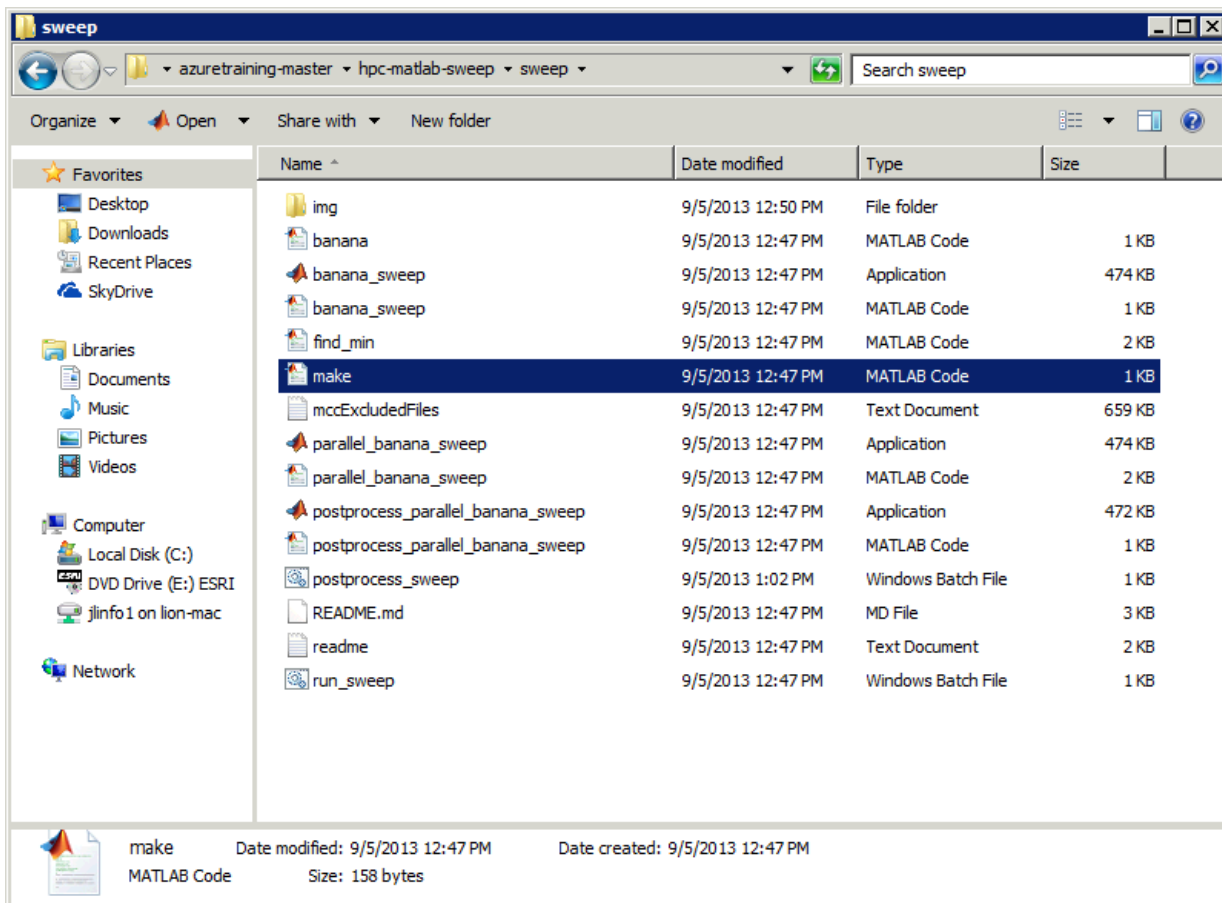
$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2.$$

It has a global minimum at $(x, y) = (1, 1)$ where $f(x, y) = 0$, but we're going to pretend we don't know that. It's called the banana function because the minimal valley looks like a banana:

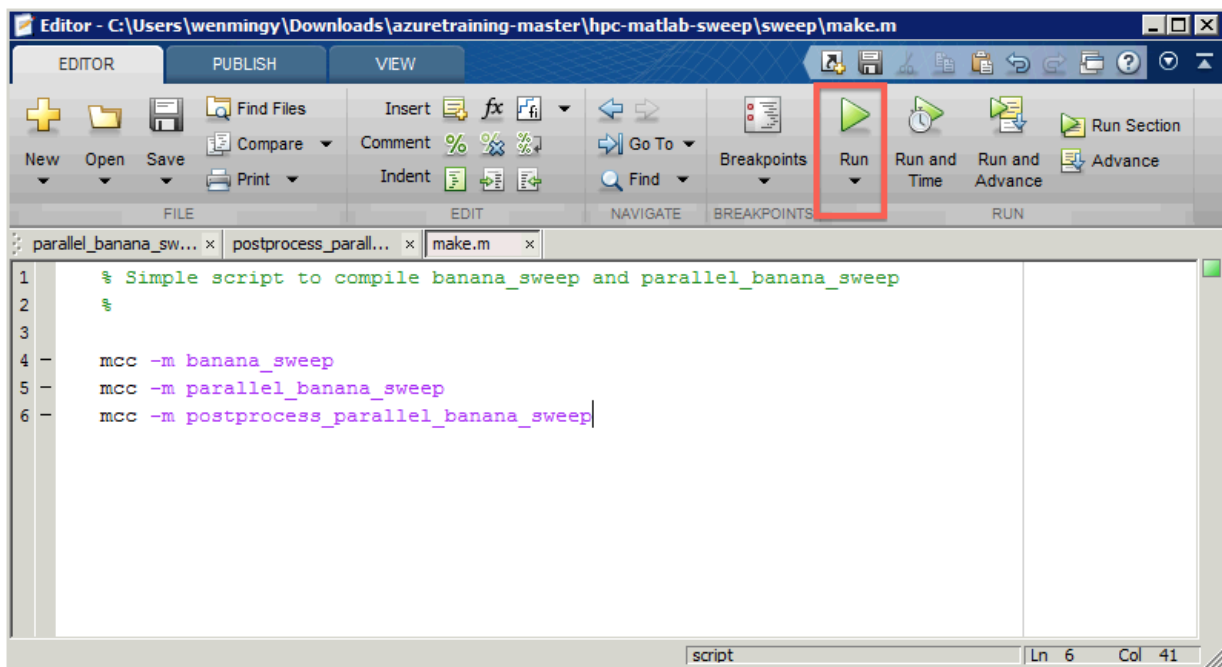


Compile MATLAB Code Into an Executable

1. Go to the \sweep folder in Explorer and double-click **make.m** to open it in the MATLAB file editor.



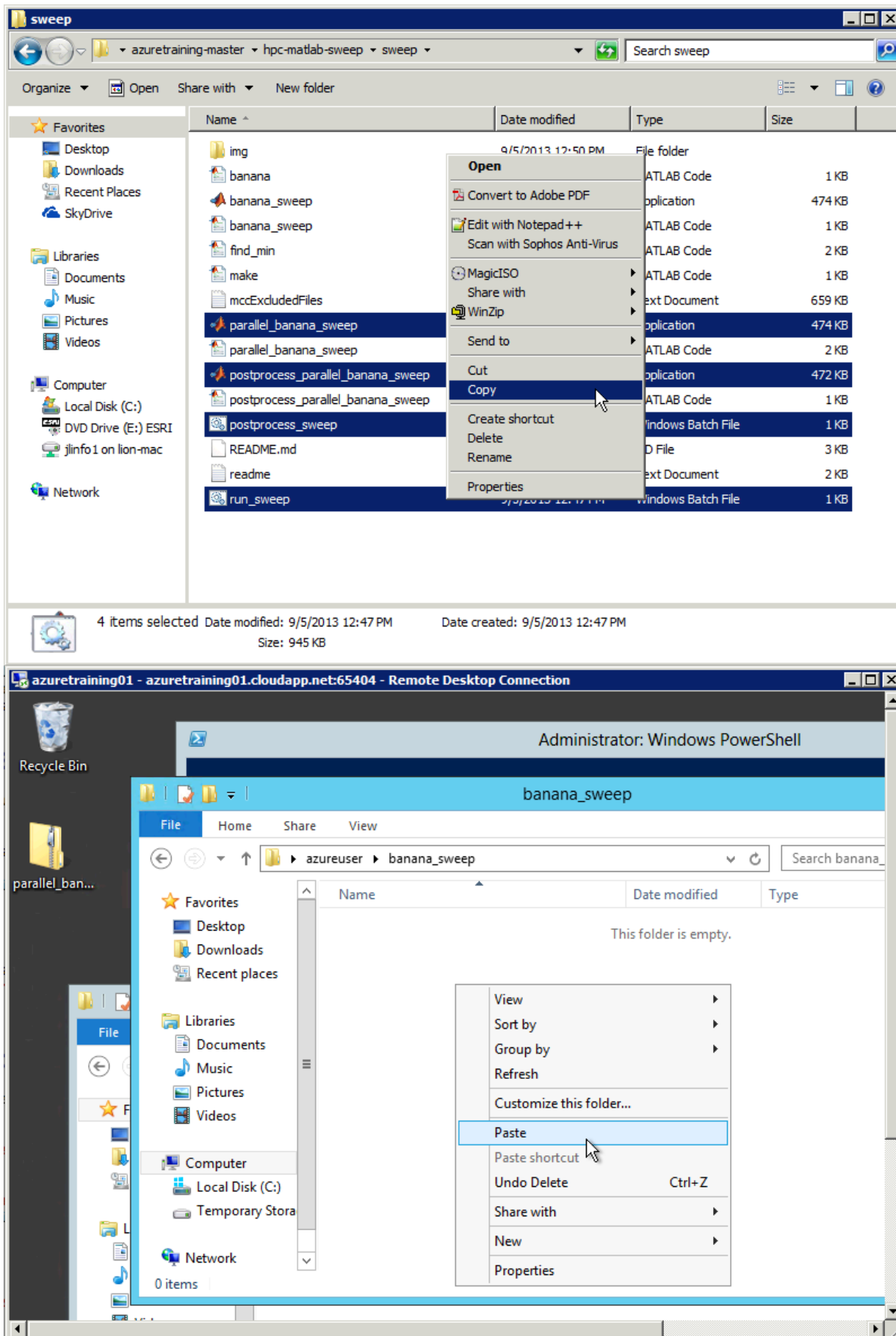
1. Click the **Run** button to execute the **make** script to produce two executables: **banana_sweep.exe** and **parallel_banana_sweep.exe**. Click **Change Folder** if you are prompted.



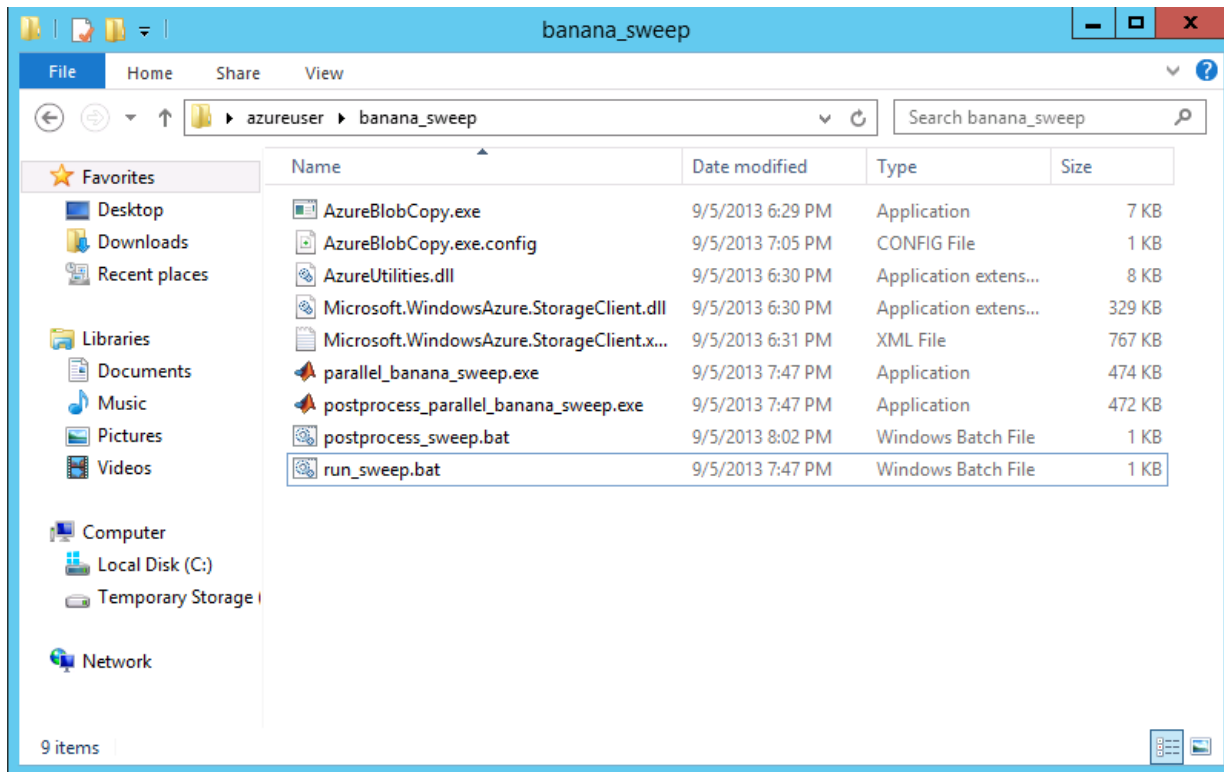
Package Parametric Sweep Executable with Required Files

1. We'll need to transfer several files to the cluster head node, so open a Remote Desktop Connection to the head node from your MATLAB workstation. Log in to the VM with the username and password you set when you created the VM, **not** the domain user you created before installing HPC Pack. This will simplify the process later on.
2. On the head node, make a folder named **banana_sweep** to contain the parametric sweep executable and supporting files.

3. Transfer **parallel_banana_sweep.exe**, **run_sweep.bat**, **postprocess_parallel_banana_sweep.exe**, and **postprocess_sweep.bat** to the **banana_sweep** folder on the cluster head node. The easiest way is to simply copy/paste the files from the workstation to the head node.



1. In the same way, transfer the contents of the **AzureBlobCopy** folder from your MATLAB workstation to **banana_sweep** on the cluster head node. The banana_sweep folder should look like this:



1. Go to the [Windows Azure Management Portal](#), click on **Storage Accounts**, select your storage account, and click on **Manage Access Keys** in the bar on the bottom. Copy the primary access key.

Windows Azure | jlinford@paratools.com

storage

NAME	STATUS	LOCATION	SUBSCRIPTION
asvworkingcluster	✓ Online	West US	Subscription-1
bigdatahackathon	✓ Online	North Europe	Subscription-1
datascience1	✓ Online	East US	Subscription-1
datasciencestorage	✓ Online	West US	Subscription-1
demo333	✓ Online	West US	Subscription-1
gutenbergc4	✓ Online	East US	Subscription-1
happycluster	✓ Online	East US	Subscription-1
hpcclusterminky	✓ Online	North Europe	Subscription-1
hpcwenming	✓ Online	North Central US	Subscription-1
joydivision	✓ Online	East US	Subscription-1
msrc	✓ Online	West US	Subscription-1
netmfdeploy1	✓ Online	West US	Subscription-1
ofcloudsc12	✓ Online	ofcloudsc12 (West US)	Subscription-1
paratools	✓ Online	paratools (West US)	Subscription-1
paraview	✓ Online	South Central US	Subscription-1
portalvhdsqdkc96d5rs0jq	✓ Online	West US	Subscription-1
pyboston	✓ Online	East US	Subscription-1
pythontutorial	✓ Online	East US	Subscription-1
websvclogs	✓ Online	North Central US	Subscription-1
wenmingy	✓ Online	North Central US	Subscription-1

MANAGE ACCESS KEYS

- Back on the cluster head node, open **banana_sweep/AzureBlobCopy.exe.config** in notepad. In that file, replace **accountName** with your storage account name and **storageKey** with your storage account's primary access key. Your AzureBlobCopy.exe.config file should look something like this:

```
<?xml version="1.0"?>
<configuration>
  <appSettings>
    <add key="StorageAccountName" value="azuretrainingstorage"/>
    <add key="StorageKey" value="ALongAndConvolutedExceptionOfRandomCharacters"/>
  </appSettings>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.0"/>
  </startup>
</configuration>
```

Package the MCR Installer

- Log in to the VM with the username and password you set when you created the VM, **not** the domain user you created before installing HPC Pack. If you log in with a domain user, the following hpcpack commands will not work.

2. We need to collect several files in order to automate the MATLAB Compiler Runtime (MCR) installation. On the cluster head node, make a folder named **MCRInstaller** in your home directory.
3. Go to <http://www.mathworks.com/products/compiler/mcr/> and download the appropriate MCR installer executable to the **MCRInstaller** folder. Download **both** the 32-bit and 64-bit versions of the package. Both 32-bit and 64-bit packages are required because our MATLAB workstation is 64-bit, but the VM is 32-bit, so both 32-bit and 64-bit libraries will be called. You can also download older versions of the MCR if you want to support older versions of MATLAB.

IMPORTANT: The MCR installation procedure for each version of MATLAB is unique.

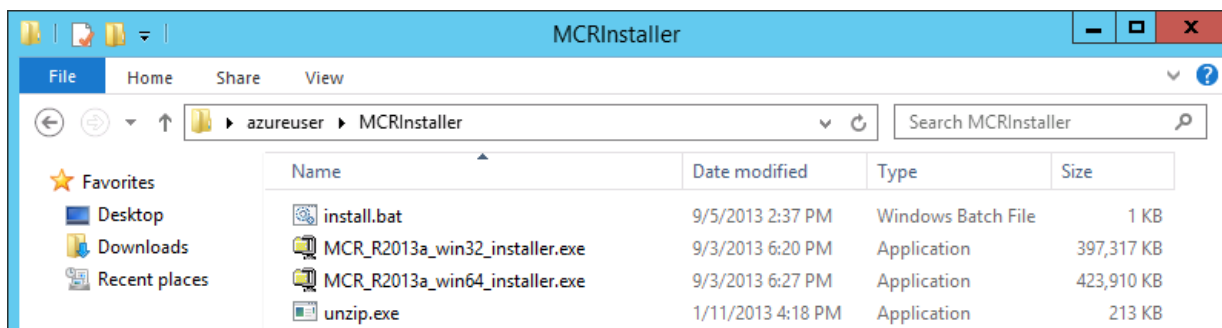
Check <http://www.mathworks.com/support> to find the installation procedure for your version. The procedure given here is for MATLAB R2013a.

1. Run all downloaded MCR installers on the head node. Run the installers and install to the default locations.
2. After the installation is complete, copy C:\Program Files (x86)\MATLAB\MATLAB Compiler Runtime\v81\bin\win32\unzip.exe to the MCRInstaller folder. We need unzip.exe to extract the MCR installers on the cluster nodes.
3. Copy all the MCR installers to the MCRInstaller folder you created earlier.
4. Open notepad and create an installation script, **install.bat**, in the MCRInstaller folder that contains these lines: mkdir win32 cd win32 ..\unzip.exe ..\MCR_R2013a\win32\installer.exe .\bin\win32\setup.exe -mode silent -agreeToLicense yes cd ..

```
mkdir win64
cd win64
..\unzip.exe ..\MCR_R2013a_win64_installer.exe
.\bin\win64\setup.exe -mode silent -agreeToLicense yes
cd ..
```

The same block of commands is repeated for the 32-bit and the 64-bit MCR installer. For each installer, the package is unzip'ed into a new subdirectory. Note that we call .\bin\winXX\setup.exe and **not** setup.exe in the package top-level. The top-level setup.exe just launches a new process so it will ignore your command line arguments. The "-agreeToLicense" argument is critical. If it is not included, the installer will hang.

5. Verify the contents of the MCRInstaller folder. It should be similar to this:



1. Open a Command Prompt window and navigate to the folder containing MCRInstaller. For example, if you created the MCRInstaller folder in your home folder, navigate to your home folder. We'll be using the `hpcpack` command to distribute MCR to the cluster nodes.
2. Use `hpcpack create` to package the MCRInstaller folder for distribution to the cluster nodes:

```
hpcpack create MCRInstaller.zip MCRInstaller
```

3. Use `hpcpack upload` to upload the package to the storage account associated with the AzureNode template you created earlier:

```
hpcpack upload MCRInstaller.zip /nodetemplate:"Default AzureNode Template" /relativePath:MCRInstaller
```

Be sure to use the **/relativePath** parameter. Otherwise the MCR installer packages will be placed on a path that involves a timestamp so it will be difficult to determine where the MCR installation files are.

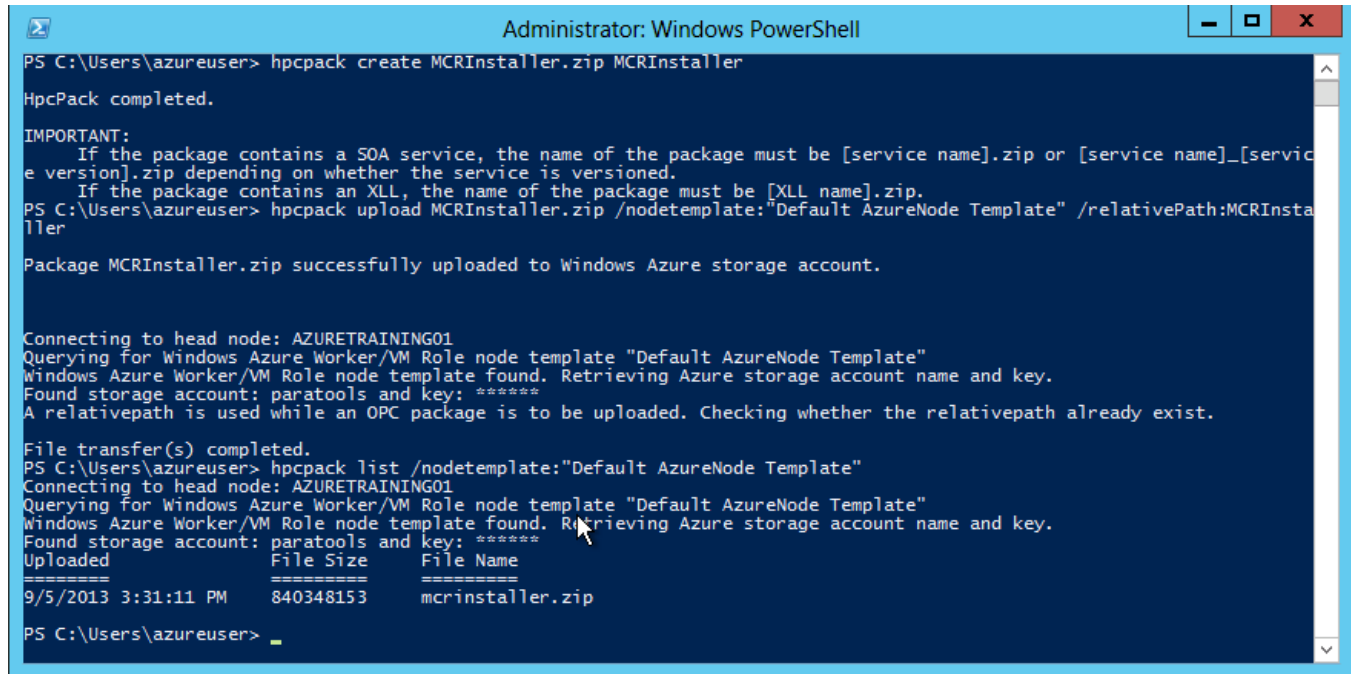
NOTE: Make sure you logged in as the right user.

If you get an "access denied" error when executing the above command, make sure you are logged in to the VM with the username and password you set when you created the VM and **not** any domain user. As a work-around, you can replace the /nodetemplate parameter with the /account and /key parameters. See the [hpcpack documentation](#) for more information.

1. Use `hpcpack list` to verify that the package is in your storage account:

```
hpcpack list /nodetemplate:"Default AzureNode Template"
```

Your console should look something like this after these commands:



```
Administrator: Windows PowerShell
PS C:\Users\azureuser> hpcpack create MCRInstaller.zip MCRInstaller
HpcPack completed.
IMPORTANT:
  If the package contains a SOA service, the name of the package must be [service name].zip or [service name].[service version].zip depending on whether the service is versioned.
  If the package contains an XLL, the name of the package must be [XLL name].zip.
PS C:\Users\azureuser> hpcpack upload MCRInstaller.zip /nodetemplate:"Default AzureNode Template" /relativePath:MCRInstaller
Package MCRInstaller.zip successfully uploaded to Windows Azure storage account.

Connecting to head node: AZURETRAINING01
Querying for Windows Azure Worker/VM Role node template "Default AzureNode Template"
Windows Azure Worker/VM Role node template found. Retrieving Azure storage account name and key.
Found storage account: paratools and key: *****
A relativepath is used while an OPC package is to be uploaded. Checking whether the relativepath already exist.

File transfer(s) completed.
PS C:\Users\azureuser> hpcpack list /nodetemplate:"Default AzureNode Template"
Connecting to head node: AZURETRAINING01
Querying for Windows Azure Worker/VM Role node template "Default AzureNode Template"
Windows Azure Worker/VM Role node template found. Retrieving Azure storage account name and key.
Found storage account: paratools and key: *****
Uploaded
=====
9/5/2013 3:31:11 PM 840348153 mcrinstaller.zip
PS C:\Users\azureuser>
```

Create a Node Startup Script

1. We'll use a startup script to automatically install MCR when the Azure nodes boot. Create **startup.bat** with the following contents:

```
cd /D %CCP_PACKAGE_ROOT%\MCRInstaller
.\install.bat
```

When the nodes boot they will use the `hpcsync` command to automatically download and unpack the MCRInstaller.zip package. By default, hpcsync deploys files to a location on the Windows Azure nodes that is determined in part by the `%CCP_PACKAGE_ROOT%` environment variable. This variable is set on Windows Azure nodes during the provisioning process. The extracted files are placed in a folder that is determined as follows: `%CCP_PACKAGE_ROOT%\`. This is the expected location for SOA services, XLLs, Excel workbooks, and startup scripts that are called from the node template. However, because we passed `"/relativePath:MCRInstaller"` to our `hpcpack upload` command, MCRInstaller.zip will be unpacked to `%CCP_PACKAGE_ROOT%\MCRInstaller`. We used `relativePath` because we cannot easily determine the part of the default path.

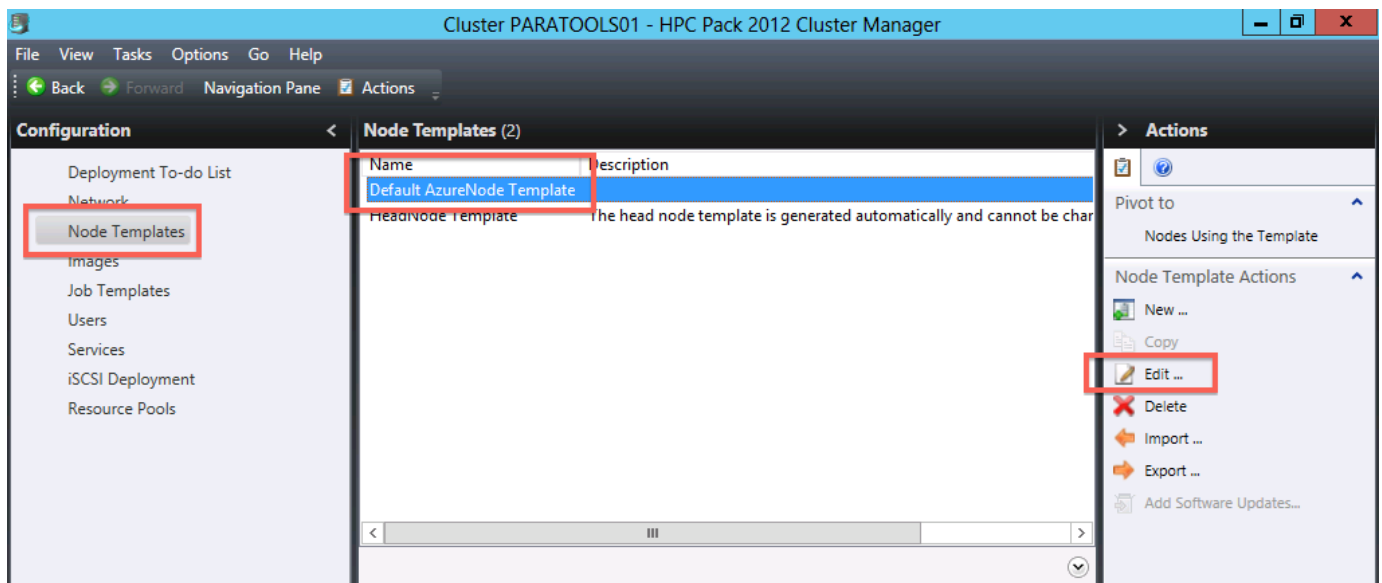
1. Open a command prompt, navigate to the folder containing startup.bat, and package the startup script:

```
hpcpack create startup.bat.zip startup.bat
```

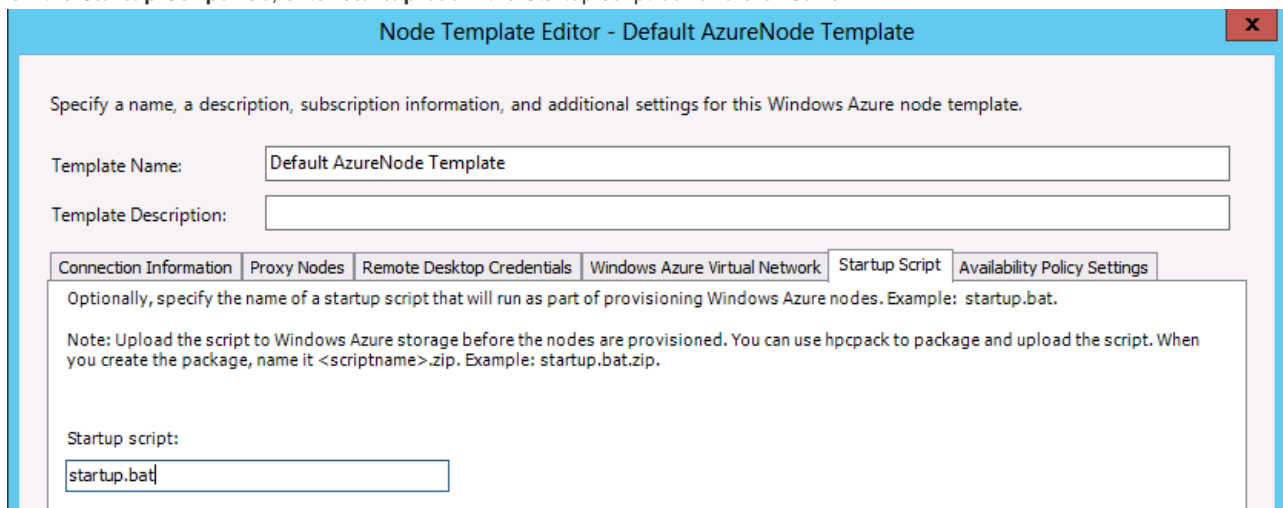
2. Upload the startup script to your storage account:

```
hpcpack upload startup.bat.zip /nodetemplate:"Default AzureNode Template"
```

3. Open the HPC Pack Cluster Manager. On the **Configuration** page, select **Node Templates** from the list on the left, select **Default AzureNode Template** from the display in the center, and click **Edit...** under Node Template Actions on the right.



- On the **Startup Script** Tab, enter **startup.bat** in the Startup script box and click **Save**.



Add and Start Cluster Nodes

- Open the Cluster Manager on the cluster head node.
- On the Node Management page, select **Add Node** under Node Actions on the right.
- Select **Add Windows Azure nodes** and click **Next**.
- Verify that **Default AzureNode Template** is the selected template, enter **4** for the number of Windows Azure nodes, and select the **Medium** node size. Click **Next** and click **Finish**.

+
Add Node Wizard
X

Specify New Nodes

Select Deployment Method

Specify New Nodes

Summary

Select a Windows Azure node template, the size of the nodes, and the number of nodes that you want to add to the cluster. Then click Next.

Windows Azure node template: Default AzureNode Template

Number of Windows Azure nodes: 4

Size of Windows Azure nodes: Medium

i

Windows Azure nodes deployed using a specific node template define a set of Windows Azure nodes. Windows Azure nodes in a set are managed as a set. You cannot start, stop, or delete individual Windows Azure nodes.

1. Select the first **AzureCN** node from the node list and click **Start** under Node Actions on the right.

Nodes (5)

List Heat Map New Tab

Node Name	Node State	Node Health	Node Template
PARATOOLS01	Online	OK	HeadNode Tem
AzureCN-0032	Not-Deployed	Unapproved	Default AzureN
AzureCN-0031	Not-Deployed	Unapproved	Default AzureN
AzureCN-0030	Not-Deployed	Unapproved	Default AzureN
AzureCN-0029	Not-Deployed	Unapproved	Default AzureN

> Actions

Pivot to

- Jobs for the Selected Nodes
- Failed Diagnostics for the Node:
- Operations for the Nodes

Node Actions

- Bring Online
- Take Offline
- Start
- Stop

You will be notified that a set of nodes is being started. Verify that four nodes will be started and click **Start**.

1. The nodes are now provisioning (this will take a while). Once the provisioning is complete, they will be in the "Unapproved" state. To approve the nodes, select **all** the nodes and click **Bring Online** under Node Actions on the right.

Nodes (5)

List Heat Map New Tab

Node Name	Node State	Node Health	Node Template
PARATOOLS01	Online	OK	HeadNode Tem
AzureCN-0032	Offline	OK	Default AzureN
AzureCN-0031	Offline	OK	Default AzureN
AzureCN-0030	Offline	OK	Default AzureN
AzureCN-0029	Offline	OK	Default AzureN

> Actions

Pivot to

- Jobs for the Selected Nodes
- Failed Diagnostics for the Node:
- Operations for the Nodes

Node Actions

- Bring Online
- Take Offline

2. Once the nodes are online you'll be able run run jobs! Note that the filesystem on the nodes is inconsistent with that of the head node. Operating system files, program files, and user files are all stored on the **D:** drive, not the C: drive as you'd expect.

Run the Parametric Sweep Job

1. Open a Command Prompt window and navigate to the folder containing the **banana_sweep** folder.
2. Use **hpcpack create** to package the executable:

```
hpcpack create banana_sweep.zip banana_sweep
```

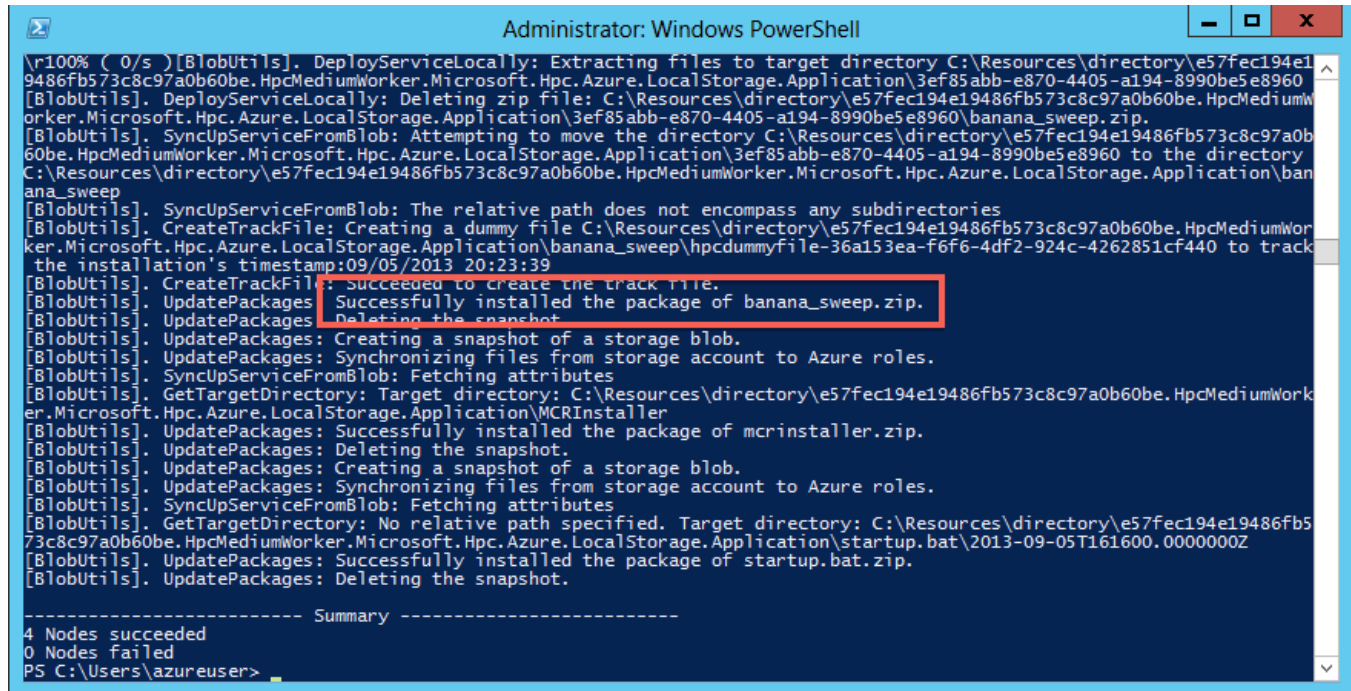
3. Use **hpcpack upload** to upload the package to your storage account. Use the `/relativePath` parameter to place `parallelbananasweep.exe` and supporting files in the **banana_sweep** folder on the compute nodes:

```
hpcpack upload banana_sweep.zip /nodetemplate:"Default AzureNode Template" /relativePath:banana_sweep
```

4. Execute **hpcsync** on all cluster nodes via the `clusrun` command. This will download the new package from your storage account to all cluster nodes:

```
clusrun /nodegroup:AzureNodes hpcsync
```

You should see output similar to this when the command completes:



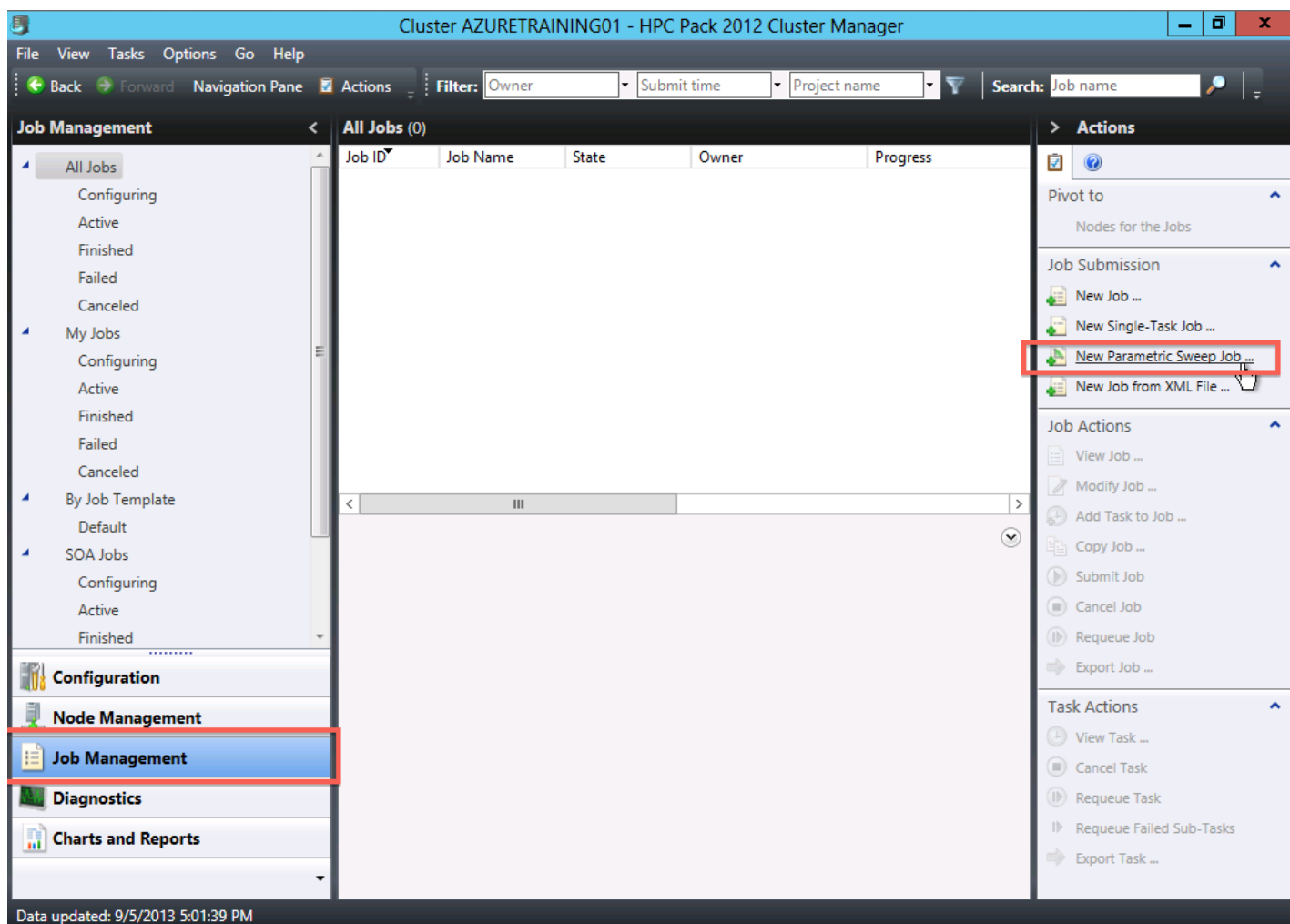
```
Administrator: Windows PowerShell

[BlobUtils]. DeployServiceLocally: Extracting files to target directory C:\Resources\directory\e57fec194e19486fb573c8c97a0b60be.HpcMediumWorker.Microsoft.Hpc.Azure.LocalStorage.Application\3ef85abb-e870-4405-a194-8990be5e8960\banana_sweep.zip.
[BlobUtils]. DeployServiceLocally: Deleting zip file: C:\Resources\directory\e57fec194e19486fb573c8c97a0b60be.HpcMediumWorker.Microsoft.Hpc.Azure.LocalStorage.Application\3ef85abb-e870-4405-a194-8990be5e8960\banana_sweep.zip.
[BlobUtils]. SyncUpServiceFromBlob: Attempting to move the directory C:\Resources\directory\e57fec194e19486fb573c8c97a0b60be.HpcMediumWorker.Microsoft.Hpc.Azure.LocalStorage.Application\3ef85abb-e870-4405-a194-8990be5e8960 to the directory C:\Resources\directory\e57fec194e19486fb573c8c97a0b60be.HpcMediumWorker.Microsoft.Hpc.Azure.LocalStorage.Application\banana_sweep
[BlobUtils]. SyncUpServiceFromBlob: The relative path does not encompass any subdirectories
[BlobUtils]. CreateTrackFile: Creating a dummy file C:\Resources\directory\e57fec194e19486fb573c8c97a0b60be.HpcMediumWorker.Microsoft.Hpc.Azure.LocalStorage.Application\banana_sweep\hpcdummyfile-36a153ea-f6f6-4df2-924c-4262851cf440 to track the installation's timestamp:09/05/2013 20:23:39
[BlobUtils]. CreateTrackFile: Succeeded to create the track file.
[BlobUtils]. UpdatePackages: Successfully installed the package of banana_sweep.zip.
[BlobUtils]. UpdatePackages: Deleting the snapshot.
[BlobUtils]. UpdatePackages: Creating a snapshot of a storage blob.
[BlobUtils]. UpdatePackages: Synchronizing files from storage account to Azure roles.
[BlobUtils]. SyncUpServiceFromBlob: Fetching attributes
[BlobUtils]. GetTargetDirectory: Target directory: C:\Resources\directory\e57fec194e19486fb573c8c97a0b60be.HpcMediumWorker.Microsoft.Hpc.Azure.LocalStorage.Application\MCRInstaller
[BlobUtils]. UpdatePackages: Successfully installed the package of mcrinstaller.zip.
[BlobUtils]. UpdatePackages: Deleting the snapshot.
[BlobUtils]. UpdatePackages: Creating a snapshot of a storage blob.
[BlobUtils]. UpdatePackages: Synchronizing files from storage account to Azure roles.
[BlobUtils]. SyncUpServiceFromBlob: Fetching attributes
[BlobUtils]. GetTargetDirectory: No relative path specified. Target directory: C:\Resources\directory\e57fec194e19486fb573c8c97a0b60be.HpcMediumWorker.Microsoft.Hpc.Azure.LocalStorage.Application\startup.bat\2013-09-05T161600.000000Z
[BlobUtils]. UpdatePackages: Successfully installed the package of startup.bat.zip.
[BlobUtils]. UpdatePackages: Deleting the snapshot.

----- Summary -----
4 Nodes succeeded
0 Nodes failed
PS C:\Users\azureuser>
```

`parallel_banana_sweep.exe` is now on the compute nodes and can be executed via a parametric sweep job.

1. Open the Cluster Manager on the head node. On the **Job Management** page, select **New Parametric Sweep Job** from the list on the right.



1. Enter **Banana Sweep Task** as the task name.

Task properties

Task name:

1. Under Step 1, set the Start Value to **1** and the End Value to **8**. We will use eight parallel jobs since we have eight CPU cores in our cluster (four medium-sized nodes, two CPU cores per node).

Step 1: Select the start and end values for the sweep task:

Start value: End value:

1. Leave the Increment Value as **1**.

Step 2: Select the amount to increment the value at each step of the sweep task:

Increment value:

1. In the command line box, enter **run_sweep.bat 0.5 -4 5 -3 6 8 ***

The command line arguments match the function parameters in the MATLAB code except that the last parameter (taskID) is given as an asterisk "***". The job scheduler replaces the asterisk with sequential integer values in the range specified in the Step 1 section, in this case, the numbers 1, 2, and so on up to 8. You can use this parameterization any way you like. In this example, we're simply using it to specify the taskID. The command line box should look like this:

Step 3: Enter the command line, working directory, and file locations for the sweep task.

Use an asterisk (*) where the step values should be inserted.

Command line: `run_sweep.bat 0.5 -4 5 -3 6 8 *`

IMPORTANT: The number of tasks is passed on the command line in this example.

If you decide to change the end value for the sweep you must update the second to last parameter on the command line.

1. Set the working directory to `%CCP_PACKAGE_ROOT%\banana_sweep`. This folder was created automatically by hpcsync because we specified the `/relativePath` parameter when we executed our hpcpack upload command.

Working directory:

`%CCP_PACKAGE_ROOT%\banana_sweep`

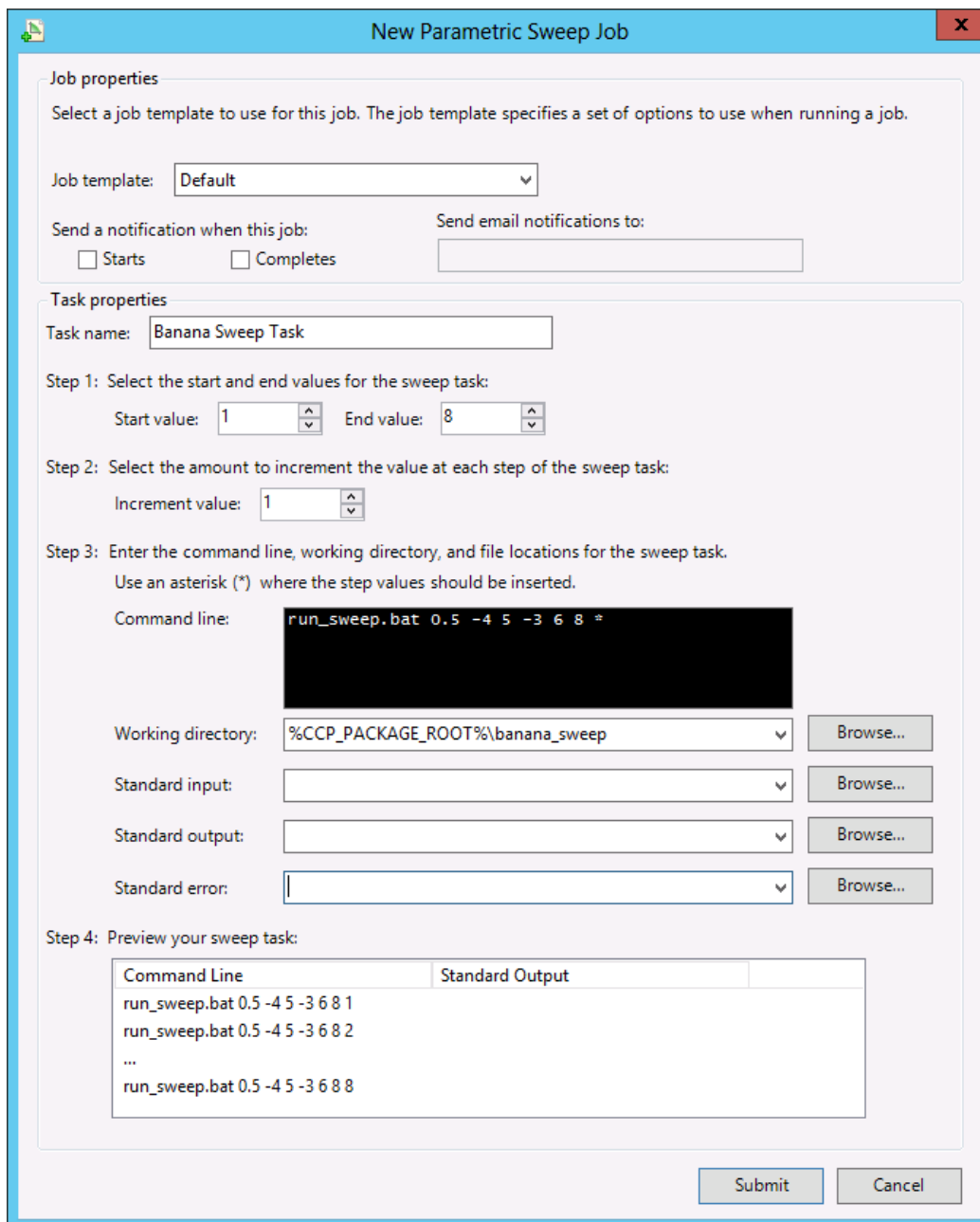


Browse...

IMPORTANT: There is a 10GB limit on `%CCP_PACKAGE_ROOT%`.

If your job is going to write a lot of data to files in the working directory then you'll need to use a different working directory and write a batch script to copy files from `%CCP_PACKAGE_ROOT%` before running the job.

1. Leave the Standard input, Standard output, and Standard error fields empty.
2. Your "New Parametric Sweep Job" window should look like this with all fields completed. Click **Submit**.



New Parametric Sweep Job

Job properties

Select a job template to use for this job. The job template specifies a set of options to use when running a job.

Job template: Default

Send a notification when this job: ☐ Starts ☐ Completes

Send email notifications to:

Task properties

Task name: Banana Sweep Task

Step 1: Select the start and end values for the sweep task:

Start value: 1 End value: 8

Step 2: Select the amount to increment the value at each step of the sweep task:

Increment value: 1

Step 3: Enter the command line, working directory, and file locations for the sweep task.
Use an asterisk (*) where the step values should be inserted.

Command line: run_sweep.bat 0.5 -4 5 -3 6 8 *

Working directory: %CCP_PACKAGE_ROOT%\banana_sweep Browse...

Standard input: Browse...

Standard output: Browse...

Standard error: Browse...

Step 4: Preview your sweep task:

Command Line	Standard Output
run_sweep.bat 0.5 -4 5 -3 6 8 1	
run_sweep.bat 0.5 -4 5 -3 6 8 2	
...	
run_sweep.bat 0.5 -4 5 -3 6 8 8	

Submit Cancel

1. You can view the job progress by selecting the job and opening the **Activity Log**:

Cluster AZURETRAINING01 - HPC Pack 2012 Cluster Manager

File View Tasks Options Go Help

Back Forward Navigation Pane Actions Filter: Owner Submit time Project name Search: Job name

Job Management

- All Jobs
 - Configuring
 - Active
 - Finished
 - Failed
 - Canceled
- My Jobs
 - Configuring
 - Active
 - Finished
 - Failed
 - Canceled
- By Job Template
 - Default
- SOA Jobs
 - Configuring
 - Active
 - Finished
 - Failed
 - Canceled
- Admin Jobs

Configuration

Node Management

Job Management

Diagnostics

Charts and Reports

Active (1)

Job ID	Job Name	State	Owner	Progress	Sub
33	Banana Sweep ...	Running	AZURETRAINING\azur...	0%	9/5

Job Name : Banana Sweep Task

Task Job Details **Activity Log**

Filter: Task name Task state Clear All

Task ID	Task Name	State	Command Line	Requester
1.1 - 1.8	Banana Sweep Task	Running	run_sweep.bat 0.5 -4 5...	1-1 Cores

As the job progresses, the allocation will be reduced on each node until the node is no longer needed.

Cluster AZURETRAINING01 - HPC Pack 2012 Cluster Manager

File View Tasks Options Go Help

Back Forward Navigation Pane Actions Filter: Owner Submit time Project name Search: Job name

Job Management

- All Jobs
 - Configuring
 - Active
 - Finished
 - Failed
 - Canceled
- My Jobs
 - Configuring
 - Active
 - Finished
 - Failed
 - Canceled
- By Job Template
 - Default
- SOA Jobs
 - Configuring
 - Active
 - Finished
 - Failed
 - Canceled
- Admin Jobs

Configuration

Node Management

Job Management

Diagnostics

Charts and Reports

Active (1)

Job ID	Job Name	State	Owner	Progress	Sub
33	Banana Sweep ...	Running	AZURETRAINING\azur...	62%	9/5/

Job Name : Banana Sweep Task

Task Job Details Activity Log

```

9/5/2013 8:37:52 PM Created by AZURETRAINING\azureuser
9/5/2013 8:58:35 PM Submitted
9/5/2013 8:58:35 PM Started
9/5/2013 8:58:35 PM Started on AZURECN-0005 with 2 cores
9/5/2013 8:58:35 PM Started on AZURECN-0006 with 2 cores
9/5/2013 8:58:35 PM Started on AZURECN-0007 with 2 cores
9/5/2013 8:58:35 PM Started on AZURECN-0008 with 2 cores
9/5/2013 8:59:05 PM Allocation reduced to 1 cores on AZURECN-0008
9/5/2013 8:59:11 PM Allocation reduced to 1 cores on AZURECN-0007
9/5/2013 8:59:11 PM Ended on AZURECN-0008
9/5/2013 8:59:18 PM Ended on AZURECN-0006
  
```

- Each task participating in the parametric sweep generated a MATLAB data file named **bananaX.mat**, where X is the taskID. run_sweep.bat used AzureBlobCopy.exe to upload the output file to a new container named **output** in the storage account specified in AzureBlobCopy.exe.config. You can see these output files in the Windows Azure Management Portal:


```
Administrator: Windows PowerShell
PS C:\Users\azureuser\banana_sweep> .\postprocess_sweep.bat 8
C:\Users\azureuser\banana_sweep>set nTasks=8
C:\Users\azureuser\banana_sweep>for /L %A IN (1 1 8) DO (AzureBlobCopy -Action Download -BlobContainer output -LocalDir . -FileName banana%A.mat )
C:\Users\azureuser\banana_sweep>(AzureBlobCopy -Action Download -BlobContainer output -LocalDir . -FileName banana1.mat )
C:\Users\azureuser\banana_sweep>(AzureBlobCopy -Action Download -BlobContainer output -LocalDir . -FileName banana2.mat )
C:\Users\azureuser\banana_sweep>(AzureBlobCopy -Action Download -BlobContainer output -LocalDir . -FileName banana3.mat )
C:\Users\azureuser\banana_sweep>(AzureBlobCopy -Action Download -BlobContainer output -LocalDir . -FileName banana4.mat )
C:\Users\azureuser\banana_sweep>(AzureBlobCopy -Action Download -BlobContainer output -LocalDir . -FileName banana5.mat )
C:\Users\azureuser\banana_sweep>(AzureBlobCopy -Action Download -BlobContainer output -LocalDir . -FileName banana6.mat )
C:\Users\azureuser\banana_sweep>(AzureBlobCopy -Action Download -BlobContainer output -LocalDir . -FileName banana7.mat )
C:\Users\azureuser\banana_sweep>(AzureBlobCopy -Action Download -BlobContainer output -LocalDir . -FileName banana8.mat )
C:\Users\azureuser\banana_sweep>postprocess_parallel_banana_sweep.exe 8
0
1.0000
1.0000
PS C:\Users\azureuser\banana_sweep> _
```

Copyright 2013 Microsoft Corporation. All rights reserved. Except where otherwise noted, these materials are licensed under the terms of the Apache License, Version 2.0. You may use it according to the license as is most appropriate for your project on a case-by-case basis. The terms of this license can be found in <http://www.apache.org/licenses/LICENSE-2.0>.