University of Manitoba

Digital Research Alliance of Canada

High Performance Computing and software environments:

Install and/or use existing software and modules

UofM-Autumn-Workshop 2022

Oct 26th-27th, 2022

Ali Kerrache

University of Manitoba

Outline

- ★ Software distribution on HPC clusters
- ★ Why modules? How to find modules?
- ★ Software stacks on Grex
- ★ How to build software from sources
  - ○ R packages
  - ○ Perl modules
  - ○ configure/make
  - ○ cmake/make
- ★ Singularity {now: apptainer}

University of Manitoba

Software distribution

Operating system package managers / repos:

- ★ **Ubuntu:** ~$ ***sudo apt-get install <package>***
- ★ **CentOS:** ~$ ***sudo yum install <package>***
- ★ **On HPC:** users do not have **sudo**! (**NO NEED TO ASK FOR IT**)

Local installation: usually to $HOME {/home/$USER} or $PROJECT

- ★ **Get the code:** download the sources/binaries: wget, git clone, … etc.
- ★ **Settings:** load dependencies, set environment variables, … etc.
- ★ **Build:** ./configure {***cmake*** ..} +opts; make; make test {***check***}; make install

Using a centralized HPC software stack:

- ★ **Software distributed via CVMFS:** CC software stack (CC clusters), …
- ★ **Local software:** modules, legally restricted software (VASP, Gaussian, …)

University of Manitoba

Software layers

User layer: Python packages, Perl and R modules, home made codes, …

Software stacks: modules for Intel, PGI, OpenMPI, CUDA, MKL, high-level applications. Multiple architectures (sse3, avx, avx2, avx512)

Nix or gentoo: GNU libc, autotools, make, bash, cat, ls, awk, grep, etc.

Gray area: Slurm, Lustre client libraries, IB/OmniPath/InfiniPath client libraries (all dependencies of OpenMPI) in Nix {or gentoo} layer, but can be overridden using PATH & LD_LIBRARY_PATH.

OS: kernel, daemons, drivers, libcuda, anything privileged (e.g. the sudo command): always local. Some legally restricted software too (VASP).

User

RSNT

Sys. Admin

University of Manitoba

Why modules?

★ Why modules?

- Control different versions of the same program.
- Avoid conflicts between different versions and libraries.
- Set the right path to each program or library.

★ Useful commands for working with modules:

- module **list**; module **avail**
- module **spider** <soft>/<version>
- module **load** soft/version; module **unload {rm}** <soft>/<version>
- module **show** soft/version; module **help** <soft>/<version>
- module **purge**; module --force **purge**
- module **use** ~/modulefiles; module **unuse** ~/modulefiles

```
[someuser@bison ]$  module list
Currently Loaded Modules:
  1) GrexEnv (S)

  Where:
   S:  Module is Sticky, requires --force to
unload or purge
```

University of Manitoba

Software stacks on Grex

- ★ Grex environment [default]: GrexEnv
  - no module loaded by default.
  - use module spider <name of the software> to search for modules
  - Compilers: {GCC, Intel}, MKL, PETSc, … etc.
  - Gaussian, ANSYS, MATLAB, … etc.
- ★ The Alliance (Compute Canada) environment [optional]: CCEnv
  - Switch to CCEnv; load a standard environment; choose the architecture[sse3, avx2, avx512], use module spider <soft>

```
module load CCEnv
module load StdEnv/2020
module load arch/avx512
module load StdEnv/2020 gcc/9.3.0 geant4/10.7.3
```

```
module load StdEnv/2016.4
module load arch/sse3
module load nixpkgs/16.09 gcc/5.4.0 geant4/10.05.p01
```

University of Manitoba

Modules on Grex

- ★ More than 500 modules:
  - GCC [5,7,9,11]; Intel [2014 - 2020].
  - Libraries: HDF5, PETSc, GSL, MKL, Libxc, Boost, ...
  - Gaussian, ANSYS, MATLAB, VASP, MCR, Java, … etc.
  - LAMMPS, GROMACS, ABINIT, QE, VMD, Molden, … etc.
- ★ Software maintenance on Grex:
  - We install programs and update modules on request from users.
  - Search for a program using "module spider <name of your program>"
  - If not installed, ask for support "support@tech.alliancecan.ca"
  - We will install the module or update the version.
  - For commercial software, contact us before you purchase the code:
    - to check license type.
    - see if it will run under Linux environment, … etc.

Modules on Grex

```
---------------------------------------------------- /opt/lmod/grex/7.6/modulefiles/Core ----------------------------------------------
  admixture/1.3.0                    intel/14.0.2.144          ncl_ncarg/6.4.0                uofm/adf/2019.305-impi
  admixture/1.23              (D)    intel/15.0.5.223          ncl_ncarg/6.5.0                uofm/adf/2020-impi
  ant/1.10.11                        intel/2017.8              nodejs/4.4.7                   uofm/adf/2020.103-impi
  circos/0.69-6                      intel/2019.5       (D)    nodejs/8.12.0                  uofm/adf/2021-impi
  cmake/3.12.3                       intel/2020.4              nodejs/13.2.0           (D)    uofm/adf/2021.102-impi
  cmake/3.14.0                       j/j903                    openbabel/2.3.2                uofm/adf/2021.106-impi    (D)
  cmake/3.16.9                       jags/3.4.0                openbabel/2.4.1                uofm/cfx/15.0
  cmake/3.23.2                (D)    jags/4.0.0                openbabel/3.0.0                uofm/cfx/16.2
  cns/1.3                            jags/4.3.0         (D)    openbabel/3.1.1         (D)    uofm/cfx/18.2
  eigen/3.3.7                        java/jdk7u25              ovito/2.9.0                    uofm/cfx/19.2
  fastqc/0.11.9                      java/jdk7u45              ovito/3.0.0-dev502      (D)    uofm/cfx/20.1
  gaussian/g09.b01.unlim             java/jdk8u5               pandoc/2.9.2.1                 uofm/cfx/20.2
  gaussian/g09.b01                   java/jdk8u66              perl/5.14.4                    uofm/cfx/21.1             (D)
  gaussian/g09.e01.unlim             java/jdk8u92       (D)    perl/5.22.1             (D)    uofm/feko/2021.2
  gaussian/g09.e01                   java/jdk13.0.1            perl/5.28.1                    uofm/gaussian/g03
  gaussian/g16.b01                   julia/1.3.0-bin           php/5.6.40                     uofm/gaussian/g09.e01
  gaussian/g16.c01.avx2.unlim        julia/1.5.4-bin           php/7.3.12              (D)    uofm/gaussian/g09         (D)
  gaussian/g16.c01.avx2              julia/1.6.1-bin           python/2.7.12-miniconda        uofm/mathematica/11.0
  gaussian/g16.c01            (D)    julia/1.7.0-bin    (D)    python/3.6-miniconda    (D)    uofm/matlab/R2014A
  gcc/4.8                            libcerf/1.4               settarg                        uofm/matlab/R2015B
  gcc/5.2                            lmod                      singularity/3.5.2              uofm/matlab/R2017A
  gcc/7.4                     (D)    ls-prepost/4.7.13         smrtlink/6.0.0.47841           uofm/matlab/R2019B
  gcc/9.2                            mcr/mcr                   stata/14.2-fagfs               uofm/matlab/R2020B2       (D)
  gcc/11.2                           mkl/10.3.11        (D)    stata/15.0-fagfs        (D)    uofm/starccm/16.06.010
  git-lfs/3.2.0                      mkl/11.1.0                tbb/14                         uofm/starccm/17.02.008-R8 (D)
  git/2.21.0                         mkl/2019.5                tbb/2019.5              (D)    uofm/umcfd/2.4
  gnuplot/5.2.7                      molden/5.9                trimmomatic/0.39               vina/1.1.2
  go/1.10.4                          multiwfn/3.8-gui          uofm/adf/2016-impi-test        vmd/1.9.3
  go/1.11.5                          multiwfn/3.8-nogui (D)    uofm/adf/2016-impi             vncworkspace/1.1
  go/1.12.12                         nbo/6.0                   uofm/adf/2017-impi             vtune/2019.4
  go/1.13                            nbo/7.0            (D)    uofm/adf/2017.114-impi         vtune/2019.5              (D)
  go/1.13.3                   (D)    ncl_ncarg/6.2.1    (D)    uofm/adf/2018dev-impi          wine/3.0
  intel/12.1.5.339                   ncl_ncarg/6.3.0           uofm/adf/2019-impi             xtb/6.5.0-bin

------------------------------------------------------ /opt/lmod/stacks -------------------------------------------------------
  CCEnv (S)    GrexEnv (S,L)
```

module avail

module spider python
module spider java

module load gcc ompi
module avail

module spider <soft>
module spider <soft>/<ver>

module show <soft>
module purge

If not available:
➔ contact support
support@tech.alliancecan.ca

University of Manitoba

Find and load Gaussian

Gaussian: restricted software; requires a registration

https://um-grex.github.io/grex-docs/docs/grex/software/specific/gaussian/

```
[someuser@bison ]$  module spider gaussian
---------------------------------------------------------------------------------------------------------------------------------------------------
  gaussian:
---------------------------------------------------------------------------------------------------------------------------------------------------
     Versions:
        gaussian/g09.e01
        gaussian/g16.b01
        gaussian/g16.c01.avx2
        gaussian/g16.c01
---------------------------------------------------------------------------------------------------------------------------------------------------
  For detailed information about a specific "gaussian" package (including how to load the modules) use the
module's full name. Note that names that have a trailing (E) are extensions provided by other modules.
  For example:
     $ module spider gaussian/g16.c01
---------------------------------------------------------------------------------------------------------------------------------------------------
```

```
[someuser@tatanka ~]$  module load gaussian/g16.c01
    Loading  Gaussian version 16.c01
```

Find and load ORCA

ORCA:

https://docs.alliancecan.ca/wiki/ORCA

- **restricted software**
- **requires a registration**

```
[someuser@bison ]$  module spider orca/5.0.2
```

```
[someuser@bison ]$  module spider orca
---------------------------------------------------------------------------------------------------------------------------------------------------------------
  orca:
---------------------------------------------------------------------------------------------------------------------------------------------------------------
     Versions:
        orca/4.2.1
        orca/5.0.1
        orca/5.0.2
---------------------------------------------------------------------------------------------------------------------------------------------------------------
  For detailed information about a specific "orca" package (including how to load the modules) use the module's full name. Note that names that have a trailing (E) are extensions provided by other modules.
  For example:
     $ module spider orca/5.0.2
---------------------------------------------------------------------------------------------------------------------------------------------------------------
```

```
[someuser@bison ]$  module load gcc/4.8  ompi/4.1.1 orca/5.0.2
    Loading module: gcc/4.8
   Loading module: ORCA/5.0.2
```

University of Manitoba

Find and load LAMMPS

```
[someuser@bison ]$  module spider lammps
----------------------------------------------------------------------------
  lammps:
----------------------------------------------------------------------------
     Versions:
        lammps/5Jun19
        lammps/5Nov16
        lammps/11Aug17
        lammps/23Jun22
        lammps/29Sep21
        lammps/30jul16
----------------------------------------------------------------------------
  For detailed information about a specific "lammps" package (including how to load the modules) use the
module's full name. Note that names that have a trailing (E) are extensions provided by other modules.
  For example:
     $ module spider lammps/29Sep21
----------------------------------------------------------------------------
```

```
[someuser@bison ]$  module spider lammps/29Sep21
```

```
[~@tatanka ~]$  module load intel/2019.5  ompi/3.1.4 lammps/29Sep21
Lmod is automatically replacing "gcc/7.4" with "intel/2019.5".
    Unloading module: gcc/7.4.0
    Loading module: LAMMPS/29Sep21
```

University of Manitoba

Building software

- Local installation (user's directory):
  - R packages; Julia packages
  - Python packages: virtual environment, conda
  - Perl modules
- Installation with:
  - make; make test {check}; make install
  - configure; make; make test {check}; make install
  - cmake; make; make test {check}; make install
- Java applications: jar files
- Singularity and/or Aptainer:
  - build the image and run the program from the container

University of Manitoba

Local installation

- R packages: minimal installation
  - R as modules: users can install the packages in their home directory.
- Python as modules: python and scipy-stack
  - users can install the packages needed in their home directory.
- Perl and bioperl as modules:
  - users can install the packages needed in their home directory.
- Other software installed locally:
  - Home made programs
  - Restricted and licensed software that can not be distributed via CVMFS.
  - Custom software: patch from a user, changing parts of the code, … etc.

https://docs.alliancecan.ca/wiki/Installing_software_in_your_home_directory

Local installation: R packages

R packages: rgdal, adegenet, stats, rjags, dplyr, … etc.

Choose a module version: module spider r

Load R and dependencies (gdal, geos, jags, gsl, udunits… etc):

```
module load gcc/7.3.0 r/3.6.0 gdal udunits
```

Launch R and install the packages:

```
~$ R
> install.packages(”sp")
'lib =/cvmfs/soft.computecanada.ca/easybuild/{..}/R/library"' is not writable
Would you like to use a personal library instead? (yes/No/cancel) yes
Would you like to create a personal library ‘~/R/{…}’ to install packages into? (yes/No/cancel) yes
--- Please select a CRAN mirror for use in this session ---
> install.packages(”dplyr")
```

University of Manitoba

Local installation: perl

Example: Hash::Merge; Logger::Simple; MCE::Mutex; threads …

Load Perl module: module load perl

Install the the first package using cpan or cpanm:

```
~$ cpan install YAML
Would you like to configure as much as possible automatically? [yes] yes
What approach do you want?  (Choose 'local::lib', 'sudo' or 'manual')
[local::lib] local::lib
Would you like me to append that to /home/$USER/.bashrc now? [yes] yes
```

Install the rest of the packages using cpan or cpanm:

```
~$ cpan install Hash::Merge
~$ cpan install Logger::Simple
~$ cpan install MCE::Mutex
```

Installation with make: STAR

- ★ Download the code {wget; curl; git clone; …}:

  wget https://github.com/alexdobin/STAR/archive/refs/tags/2.7.8a.tar.gz

- ★ Unpack the code: tar -xvf 2.7.8a.tar.gz
- ★ Load GCC compiler: module load gcc
- ★ Compile the code:

  cd STAR-2.7.8a/source

  make

~/.bash_profile

PATH=$PATH:$HOME/bin:$HOME/software/star/2.7.8a/bin
export PATH

- ★ Copy the binaries and set the path:

  mkdir -p ~/software/star/2.7.8a/bin

  cp STAR ~/software/star/2.7.8a/bin

  export PATH=$PATH:~/software/star/2.7.8a/bin

University of Manitoba

Installation with configure/make

- ★ Download and unpack the code: wget, … gunzip, … etc.
- ★ Load the modules and dependencies: module load gcc ompi fftw
- ★ Configure the program
  - ○ If configure not included, run: autoreconf -fvi [to generate it].
  - ○ ./configure --help [to see the different options].
  - ○ ./configure --prefix=<path to install dir> {+other options}
- ★ Compile and test:
  - ○ make; make -j4
  - ○ make check; make test
- ★ Install the program:
  - ○ make install

University of Manitoba

Example: PETSc

```
./configure --with-blas-lapack-dir=$MKLROOT/lib/intel64  --prefix=${instdir}  --with-cxx-dialect=C++11
--download-scalapack=yes --download-blacs=yes  --download-superlu_dist=yes
--download-mumps=yes --download-parmetis=yes --download-metis=yes --download-spooles=yes
--download-cproto=yes --download-prometheus=yes --with-mkl_pardiso=1
--with-mkl_pardiso-dir=$MKLROOT --with-mkl-sparse-optimize=1  --with-scalar-type=complex
--with-debugging=0  --with-hdf5=yes --with-hdf5-dir=$HDF5HOME  --download-suitesparse=yes
--download-fftw=${fftsrc} --download-amd=yes --download-adifor=yes  --download-superlu=yes
--download-triangle=yes --download-generator=yes  --with-64-bit-pointers=no  --with-cc=mpicc
--CFLAGS='-O2  -msse4.2 -xSSE4.2 -mp1 -I$MKLROOT/include -mkl -fPIC '  --with-cxx='mpicxx'
--CXXFLAGS='-O2 -msse4.2 -xSSE4.2 -mp1 -I$MKLROOT/include -mkl -std=c++11 -fPIC '
--with-fc='mpif90'  --FFLAGS='-O2 -msse4.2 -xSSE4.2 -mp1 -I$MKLROOT/include -mkl -fPIC '
--with-single-library=yes --with-shared-libraries=yes  --with-shared-ld=mpicc
--sharedLibraryFlags="-fpic -mkl -fPIC"  --with-mpi=yes  --with-mpi-shared=yes --with-mpirun=mpiexec
--with-mpi-compilers=yes  --with-x=yes
```

University of Manitoba

Example: ABINIT

```
module load intel ompi gsl netcdf
instdir=<path to the installation directory>

../configure --prefix=${instdir} --enable-mpi --enable-mpi-io --with-fft-flavor=fftw3-mkl
--with-linalg-flavor=mkl --with-math-flavor=gsl --enable-debug="no"
--enable-optim="standard" --enable-64bit-flags --with-linalg-libs="-L$MKLROOT/lib/intel64
-lmkl_scalapack_lp64 -lmkl_blacs_openmpi_lp64  -lmkl_intel_lp64 -lmkl_sequential
-lmkl_core -lm"  --with-fft-incs="-I$MKLROOT/include/fftw -I$MKLROOT/interfaces/fftw3xf"
--with-fft-libs="-L$MKLROOT/interfaces/fftw3xf -lfftw3xf_intel_lp64"
--with-dft-flavor="atompaw+libxc+wannier90" --with-trio-flavor="netcdf"  --enable-lotf
--enable-macroave   --enable-gw-dpc  CC=mpicc CXX=mpic++ FC=mpif90 F77=mpif77
F90=mpif90
```

University of Manitoba

Example with cmake/make

- ★ Download and unpack the code: wget, … gunzip, … etc.
- ★ Load the modules and dependencies: module load gcc ompi fftw
- ★ Configure the program: you may need to load cmake module
  - mkdir build && cd build
  - cmake .. --help [to see the different options].
  - cmake .. -DCMAKE_INSTALL_PREFIX=installdir {+other options}
- ★ Compile and test:
  - make; make -j8
  - make check; make test
- ★ Install the program:
  - make install

Using the GUI:

ccmake .. {+ options}

University of Manitoba

Cmake options for GROMACS

- Download and unpack the source files
- Load modules:

  module load intel/15.0 ompi/3.1.4 fftw cmake
- configure; compile; install

  cd gromacs-5.1.4; mkdir build; cd build

  cmake -DCMAKE_INSTALL_PREFIX=<path to install dir> -DBUILD_SHARED_LIBS=off -DBUILD_TESTING=off -DREGRESSIONTEST_DOWNLOAD=off -DCMAKE_C_COMPILER=`which mpicc` -DCMAKE_CXX_COMPILER=`which mpicxx` -DGMX_BUILD_OWN_FFTW=on -DGMX_SIMD=SSE4.1 -DGMX_DOUBLE=off -DGMX_EXTERNAL_BLAS=on -DGMX_EXTERNAL_LAPACK=on -DGMX_FFT_LIBRARY=fftw3 -DGMX_GPU=off -DGMX_MPI=on -DGMX_OPENMP=off -DGMX_X11=on ../gromacs-5.1.4

  make -j4; make install

Cmake options for CERN-ROOT

- ★ Download and unpack the source files
- ★ Load modules:

  module load gcc/9.2 tbb/2019.5 gsl/2.7.1 python/3.7.5 fftw/3.3.8

  module load cfitsio/4.0.0 mkl/2019.5 cmake/3.16.9 git

  export FFTW_DIR=/global/software/cent7/fftw/3.3.8-gcc92-ompi314
- ★ configure; compile; install

```
mkdir build && cd build
cmake -DCMAKE_INSTALL_PREFIX=<path to install dir>  -DGCC_INSTALL_PREFIX=/global/software/cent7/gcc-9.2.0-rpath-76/ \
-DPYTHON_EXECUTABLE=`which python` -DGSL_CONFIG_EXECUTABLE=/global/software/cent7/gsl/2.7.1-gcc92/bin/gsl-config \
-DGSL_INCLUDE_DIR=/global/software/cent7/gsl/2.7.1-gcc92/include
-DGSL_LIBRARY=/global/software/cent7/gsl/2.7.1-gcc92/lib/libgsl.so \
-DGSL_ROOT_DIR=/global/software/cent7/gsl/2.7.1-gcc92/  -DCMAKE_C_COMPILER=`which gcc` \
-DCMAKE_CXX_COMPILER=`which g++` -DCMAKE_FORTRAN_COMPIER=`which gfortran` \
-DCFITSIO_INCLUDE_DIR=/global/software/cent7/cfitsio/4.0.0-gcc92/include \
-DCFITSIO_LIBRARY=/global/software/cent7/cfitsio/4.0.0-gcc92/lib/libcfitsio.so \
-DJEMALLOC_ROOT_DIR=/usr  -Dpython3=ON -Dpcre=ON -Dzlib=ON Dunuran=ON -Dexplicitlink=ON -Dminuit2=ON -Droofit=ON \
-DCMAKE_SKIP_RPATH=ON -Dxrootd=OFF -Dmysql=OFF -Dkrb5=OFF -Dodbc=OFF -Doracle=OFF \
-DQT_QMAKE_EXECUTABLE=/usr/bin/qmake-qt4  <path to source directory>
```

University of Manitoba

Java applications

- ★ Download and unpack the code
- ★ Load java module: module load java
- ★ Run the code

- ★ Example: Trimmomatic
  - wget http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/Trimmomatic-0.39.zip
  - unzip Trimmomatic-0.39.zip

- ★ Run the code

```
module load java
java -jar <path to>/trimmomatic-0.39.jar {+options if any}
```

University of Manitoba

Singularity

Resources: Github, DockerHub, SingularityHub, Aptainer.

Singularity examples: https://github.com/singularityware/singularity/tree/master/examples

- ★ **Documentation:** http://singularity.lbl.gov/user-guide
- ★ **DockerHub:** https://hub.docker.com/explore/
- ★ **SingularityHub:** https://www.singularity-hub.org/
- ★ **Aptainer:** https://apptainer.org/

https://docs.alliancecan.ca/wiki/Singularity/en
https://um-grex.github.io/grex-docs/

Access to Singularity:

- ★ **Connect to cluster:** Grex, cedar, graham, beluga or narval:
- ★ **Load a module:** module load singularity
- ★ **Build the image:** convert the image from Docker to Singularity
- ★ **Note:** You may need to use your own Linux machine or VM to build the image

University of Manitoba

Singularity

- ★ Alternative for running software: difficult to build from source
- ★ Possibilite to convert Docker images to singularity.
- ★ Singularity installed on all clusters {no Docker for security reasons}
- ★ Build the image:

```
module load singularity
singularity build qiime2-2021.11.sif docker://quay.io/qiime2/core:2021.11
```

- ★ Run the code via singularity:

```
singularity exec -B $PWD:/home -B /global/scratch/someuser:/outputs  \
-B /global/scratch/someuser/path/to/inputs:/inputs <path to qiime2-2021.11.sif>   \
qiime feature-classifier fit-classifier-naive-bayes \
--i-reference-reads  /outputs/some_output_feature.qza \
--i-reference-taxonomy /outputs/some_output_ref-taxonomy.qza \
--o-classifier /outputs/some_output_classifier.qza
```

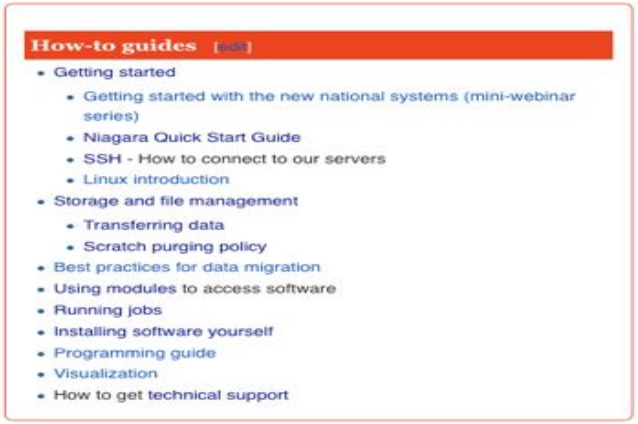
The Alliance [Compute Canada] wiki

- Systems and services
- Guides

- Links to specific documentation by disciplines
- Links to the documentation from regional partners

Systems and services [edit]

- List of current Compute Canada systems
- Cedar, Graham and Béluga, general-purpose clusters
  - System status and **upcoming outages**
  - Known issues
- Niagara, a cluster designed for large parallel jobs
- Hélios, a GPU cluster
- Available software
- National Data Cyberinfrastructure, long-term and tape storage services (limited availability)
- Cloud computing service
- Globus file transfer service
- Policy table of contents
- FAQ, Frequently Asked Questions
- Using a resource allocation, a guide for Principal Investigators
  - RAC 2019 transition FAQ, notes on the implementation of 2019 RAC awards



How-to guides [edit]

- Getting started
  - Getting started with the new national systems (mini-webinar series)
  - Niagara Quick Start Guide
  - SSH - How to connect to our servers
  - Linux introduction
- Storage and file management
  - Transferring data
  - Scratch purging policy
- Best practices for data migration
- Using modules to access software
- Running jobs
- Installing software yourself
- Programming guide
- Visualization
- How to get technical support

Discipline guides [edit]

- AI and Machine Learning
- Bioinformatics
- Biomolecular simulation
- Computational chemistry
- Computational fluid dynamics (CFD)
- Geographic information systems (GIS)
- Humanities
- Subatomic physics

Regional partners and services [edit]

- WestGrid
- SHARCNET
- SciNet
- Centre for Advanced Computing
- Calcul Québec
- ACENET
- ownCloud storage service

Grex documentation

Unofficial Grex User Guide

Search

Notes of Grex Changes

Accessing Compute Canada resources

Grex HPC Documentation

Access and Usage conditions

Connecting / Transferring data

Storage and Data

Running Jobs

Software

Frequently Asked Questions

Local IT Resources

Support and Training

Disclaimer

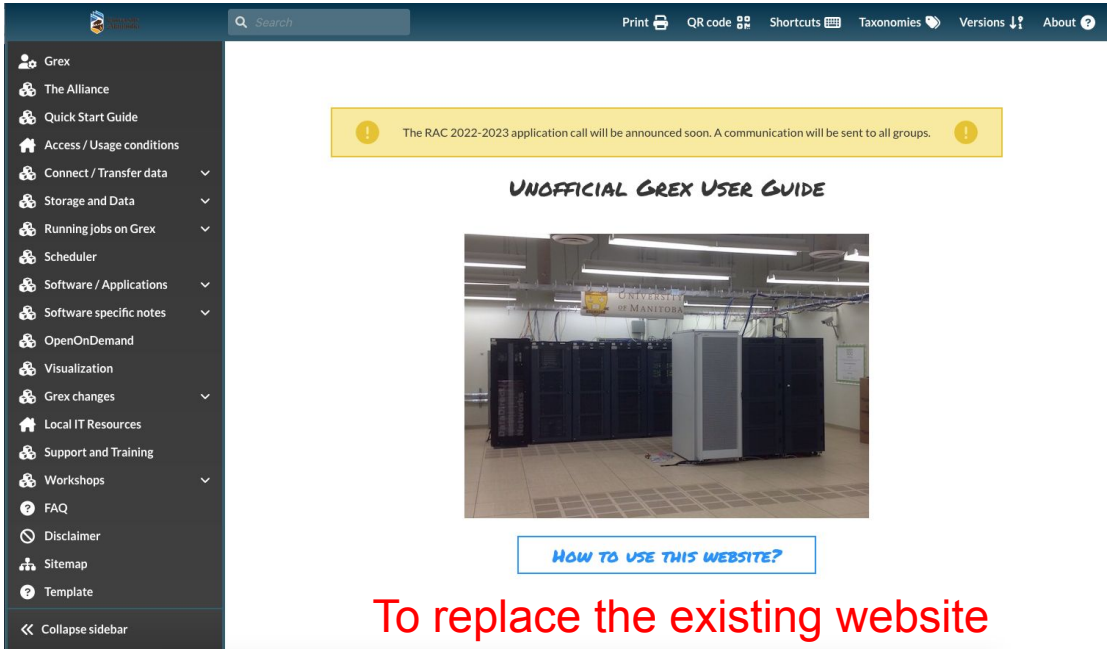
User documentation for HPC resources at University of Manitoba

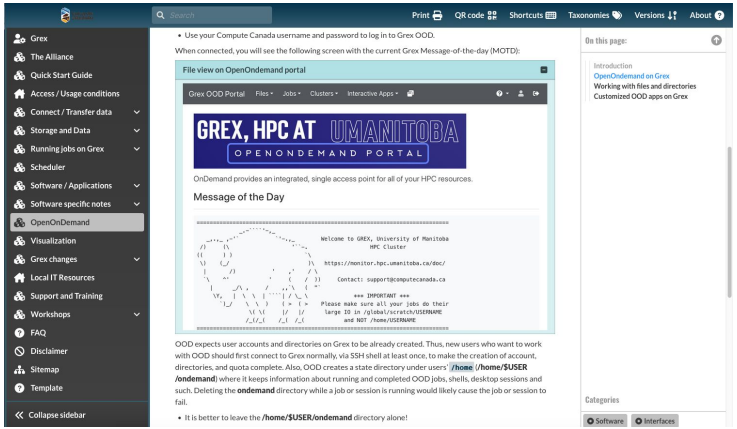
Since you have found this Website, you may be interested in Grex documentation. Grex is the University of Manitoba's High-Performance Computing system.

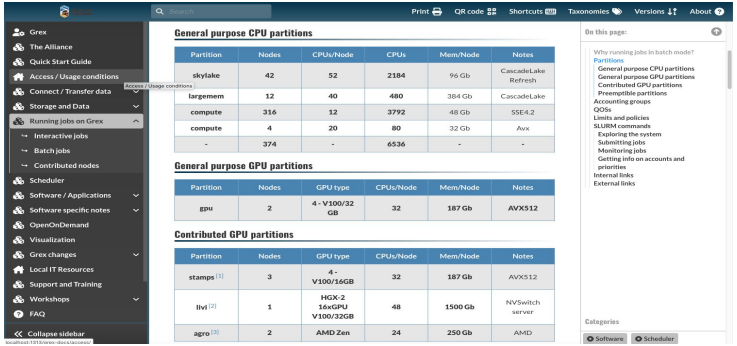
- Updating the documentation after adding the new hardware.
- Hosted on GitHub
- The link is available as MOTD when login to Grex.

https://um-grex.github.io/grex-docs/

Grex documentation: new website







To replace the existing website

https://um-grex.github.io/grex-docs/

Online soon !

University of Manitoba

Useful links

- ★ FAQ: https://docs.alliancecan.ca/wiki/Frequently_Asked_Questions
- ★ Jobs:
  - https://docs.alliancecan.ca/wiki/Running_jobs
  - https://docs.alliancecan.ca/wiki/Job_scheduling_policies
  - https://docs.alliancecan.ca/wiki/Advanced_MPI_scheduling
  - https://docs.alliancecan.ca/wiki/Using_GPUs_with_Slurm
- ★ Storage:
  - https://docs.alliancecan.ca/wiki/Storage_and_file_management
  - https://docs.alliancecan.ca/wiki/Transferring_data/en
- ★ Software:
  - https://docs.alliancecan.ca/wiki/Available_software
  - https://docs.alliancecan.ca/wiki/Utiliser_des_modules/en
  - https://docs.alliancecan.ca/wiki/Installing_software_in_your_home_directory
- ★ Grex: https://um-grex.github.io/grex-docs/

Upcoming WestDRI trainings

https://training.westdri.ca/

https://autumnschool2022.westdri.ca/info/