# 2 Lecture 2:Jan 22

## Last time

- Introduction
- Course logistics

## Today

- Introduce yourself (remind remote students to record a short video)
  - basic info (name, department, year, ...)
  - why taking this course
- Git
- Linear algebra: vector and vector space, rank of a matrix

## What is git?

Git is currently the most popular system for version control according to Google Trend. Git was initially designed and developed by Linus Torvalds in 2005 for Linux kernel development. Git is the British English slang for unpleasant person.

## Why using git?

- GitHub is becoming a de facto central repository for open source development.
- **Advertise** yourself through GitHub (e.g., host a free personal webpage on GitHub).
- a skill that employers look for (according to this AmStat article).

## Git workflow

Figure 1 shows its basic workflow.

## What do I need to use Git?

- A **Git server** enabling multi-person collaboration through a centralized repository.
- A **Git client** on your own machine.
  - Linux: Git client program is shipped with many Linux distributions, e.g., Ubuntu and CentOS. If not, install using a package manager, e.g., `yum install git` on CentOS.
  - Mac: follow instructions at https://www.atlassian.com/git/tutorials/install-git.

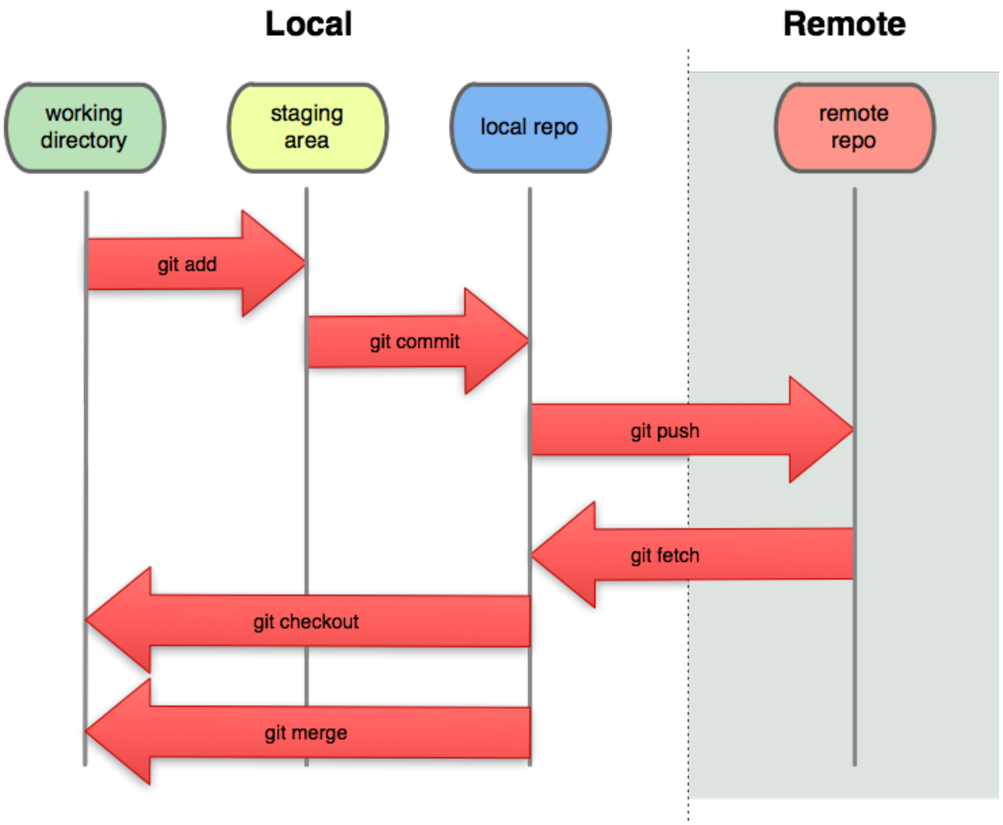

Figure 1

  - Windows: Git for Windows at https://gitforwindows.org (GUI) aka Git Bash.

- Do **not** totally rely on GUI or IDE. Learn to use Git on command line, which is needed for cluster and cloud computing.

## Git survival commands

- `git pull` synchronize local Git directory with remote repository.
- Modify files in local working directory.
- `git add FILES` add snapshots to staging area
- `git commit -m “message”` store snapshots permanently to (**local**) Git repository
- `git push` push commits to remote repository.

## Git basic usage

Working with your local copy.

- `git pull` : update local Git repository with remote repository (fetch + merge).
- `git log FILENAME` : display the current status of working directory.
- `git diff` : show differences (by default difference from the most recent commit).
- `git add file1 file2 ...` : add file(s) to the staging area.
- `git commit` : commit changes in staging area to Git directory.
- `git push` : publish commits in local Git repository to remote repository.
- `git reset –soft HEAD 1` : undo the last commit.
- `git checkout FILENAME` : go back to the last commit, discarding all changes made.
- `git rm FILENAME` : remove files from git control.

## Vector and vector space

(from JM Appendix A)

- A set of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are *linearly dependent* if there exist coefficients $c_j$ for $j = 1, 2, \ldots, n$ such that $\sum_{j=1}^n c_j \mathbf{x}_j = \mathbf{0}$ and $||\mathbf{c}||_2 = \sum_{j=1}^n c_j^2 > 0$. They are *linearly independent* if $\sum_{j=1}^n c_j \mathbf{x}_j = \mathbf{0}$ implies $c_j = 0$ for all $j$.
- Two vectors are *orthogonal* to each other, written $\mathbf{x} \perp \mathbf{y}$, if their inner product is 0, that is $\mathbf{x}^T \mathbf{y} = \mathbf{y}^T \mathbf{x} = \sum\limits_j x_j y_j = 0$.
- A set of vectors $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}$ are mutually orthogonal iff $\mathbf{x}^{(i)T} \mathbf{x}^{(j)} = 0$ for $\forall i \neq j$.
- The most common set of vectors that are mutually orthogonal are the *elementary* vectors $\mathbf{e}^{(1)}, \mathbf{e}^{(2)}, \ldots, \mathbf{e}^{(n)}$, which are all zero, except for one element equal to 1, so that $\mathbf{e}_i^{(i)} = 1$ and $\mathbf{e}_j^{(i)} = 0, \forall j \neq i$.
- A *vector space* $\mathcal{S}$ is a set of vectors that are closed under addition and scalar multiplication, that is
  - if $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are in $\mathcal{S}$, then $c_1 \mathbf{x}^{(1)} + c_2 \mathbf{x}^{(2)}$ is in $\mathcal{S}$.
- A vector space $\mathcal{S}$ is *generated* or *spanned* by a set of vectors $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}$, written as $\mathcal{S} = \text{span}\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}\}$, if any vector $\mathbf{x}$ in the vector space is a linear combination of $\mathbf{x}_i, i = 1, 2, \ldots, n$.
- A set of linearly independent vectors that generate or span a space $\mathcal{S}$ is called a *basis* of $\mathcal{S}$.

### Example A.1

Let

$$\mathbf{x}^{(1)} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \mathbf{x}^{(2)} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}, \text{ and } \mathbf{x}^{(3)} = \begin{bmatrix} -3 \\ -1 \\ 1 \\ 3 \end{bmatrix}.$$

Then $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are linearly independent, but $\mathbf{x}^{(1)}$, $\mathbf{x}^{(2)}$, and $\mathbf{x}^{(3)}$ are linearly dependent since $5\mathbf{x}^{(1)} - 2\mathbf{x}^{(2)} + \mathbf{x}^{(3)} = 0$

## Rank

Some matrix concepts arise from viewing columns or rows of the matrix as vectors. Assume $\mathbf{A} \in \mathbb{R}^{m \times n}$.

- rank($\mathbf{A}$) is the maximum number of linearly independent rows or columns of a matrix.
- rank($\mathbf{A}$) $\leqslant \min\{m, n\}$.

- A matrix is *full* rank if $\text{rank}(\mathbf{A}) = \min\{m, n\}$. It is *full row rank* if $\text{rank}(\mathbf{A}) = m$. It is *full column rank* if $\text{rank}(\mathbf{A}) = n$.

- a square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is *singular* if $\text{rank}(\mathbf{A}) < n$ and *non-singular* if $\text{rank}(\mathbf{A}) = n$.

- $\text{rank}(\mathbf{A}) = \text{rank}(\mathbf{A}^T) = \text{rank}(\mathbf{A}^T\mathbf{A}) = \text{rank}(\mathbf{A}\mathbf{A}^T)$. (Show this in HW.)

- $\text{rank}(\mathbf{AB}) \leqslant \min\{\mathbf{A}, \mathbf{B}\}$. (Hint: Columns of **AB** are spanned by columns of A and rows of of **AB** are spanned by rows of B.)

- if $\mathbf{Ax} = \mathbf{0}_m$ for some $\mathbf{x} \neq \mathbf{0}_n$, then $\text{rank}(\mathbf{A}) \leqslant n - 1$.