Part II: Least Squares

LS Solution

Theorem 1. A vector $\mathbf{x}_{\mathsf{LS}}$ is an optimal solution to the LS problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2$$

if and only if it satisfies

$$\mathbf{A}^T\mathbf{A}\mathbf{x}_{\mathsf{LS}} = \mathbf{A}^T\mathbf{y}. \tag{$*$}$$

- the optimality condition in $(*)$ is true for any $\mathbf{A}$, not just full-column rank $\mathbf{A}$
- suppose that $\mathbf{A}$ has full-column rank
  - $(*)$ is a symmetric PD linear system
  - the Gram matrix $\mathbf{A}^T\mathbf{A}$ is nonsingular (easy to verify)
  - the solution to $(*)$ is uniquely given by $\mathbf{x}_{\mathsf{LS}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y}$
- $(*)$ is called the normal equations
- the same result holds for the complex case, viz., $\mathbf{A}^H\mathbf{A}\mathbf{x}_{\mathsf{LS}} = \mathbf{A}^H\mathbf{y}$
- LS is a unconstrained optimization problem with a quadratic objective
- Linear regression is a linear approach to modelling the relationship between a dependent variable and one or more independent variables (i.e., data fitting) which can be estimated using LS method (or based LS loss function).

LS Solution

- there are many ways to prove Theorem 1, such as by the projection theorem, by optimization, or by singular value decomposition
  - projection theorem
  - optimization
  - singular value decomposition (cf. Singular Value Decomposition Topic)
  - more...

LS and Projection Theorem

- Theorem 1 can be shown using the projection theorem
- let $\mathbf{x}_{\text{LS}}$ be an LS solution, and observe that

$$\Pi_{\mathcal{R}(\mathbf{A})}(\mathbf{y}) = \arg\min_{\mathbf{z}\in\mathcal{R}(\mathbf{A})} \|\mathbf{z}-\mathbf{y}\|_2^2 = \mathbf{A}\mathbf{x}_{\text{LS}}$$

- by the projection theorem (Theorem 2 in Basic Concepts Topic), we have

$$\begin{aligned}
\Pi_{\mathcal{R}(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\mathbf{x}_{\text{LS}} \quad &\Longleftrightarrow \quad \mathbf{z}^T(\mathbf{A}\mathbf{x}_{\text{LS}} - \mathbf{y}) = 0 \text{ for all } \mathbf{z} \in \mathcal{R}(\mathbf{A}) \\
&\Longleftrightarrow \quad \mathbf{x}^T\mathbf{A}^T(\mathbf{A}\mathbf{x}_{\text{LS}} - \mathbf{y}) = 0 \text{ for all } \mathbf{x} \in \mathbb{R}^n \\
&\Longleftrightarrow \quad \mathbf{A}^T(\mathbf{A}\mathbf{x}_{\text{LS}} - \mathbf{y}) = \mathbf{0} \\
&\Longleftrightarrow \quad \mathbf{A}^T\mathbf{A}\mathbf{x}_{\text{LS}} = \mathbf{A}^T\mathbf{y}
\end{aligned}$$

Orthogonal Projections

- the projections of $\mathbf{y}$ onto $\mathcal{R}(\mathbf{A})$ and $\mathcal{R}(\mathbf{A})^\perp$ (for full column-rank $\mathbf{A}$) are, resp.,

$$\Pi_{\mathcal{R}(\mathbf{A})}(\mathbf{y}) = \mathbf{A}\mathbf{x}_{\mathsf{LS}} = \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y}$$

$$\Pi_{\mathcal{R}(\mathbf{A})^\perp}(\mathbf{y}) = \mathbf{y} - \Pi_{\mathcal{R}(\mathbf{A})}(\mathbf{y}) = (\mathbf{I} - \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T)\mathbf{y}$$

- the orthogonal projector of $\mathbf{A}$ is defined as

$$\mathbf{P}_\mathbf{A} = \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$$

  the orthogonal complement projector of $\mathbf{A}$ is defined as

$$\mathbf{P}_\mathbf{A}^\perp = \mathbf{I} - \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T = \mathbf{I} - \mathbf{P}_\mathbf{A}.$$

- obviously, we want to write $\Pi_{\mathcal{R}(\mathbf{A})}(\mathbf{y}) = \mathbf{P}_\mathbf{A}\mathbf{y}$, $\Pi_{\mathcal{R}(\mathbf{A})^\perp}(\mathbf{y}) = \mathbf{P}_\mathbf{A}^\perp\mathbf{y}$
- note: a more general definition for orthogonal projectors for general $\mathbf{A}$ will be studied in Singular Value Decomposition Topic

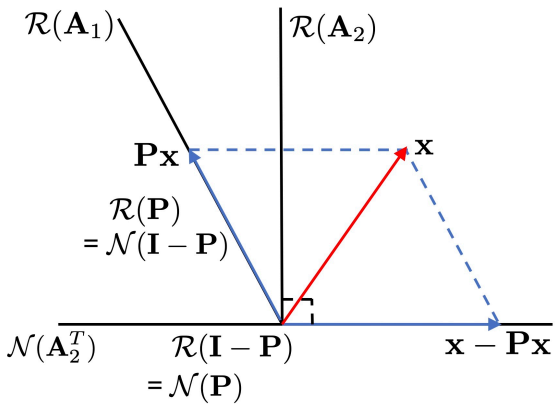
Orthogonal Projections

- properties of $\mathbf{P}_{\mathbf{A}}$ (same properties apply to $\mathbf{P}_{\mathbf{A}}^{\perp}$):
  - $\mathbf{P}_{\mathbf{A}}$ is idempotent; i.e., $\mathbf{P}_{\mathbf{A}}^2 = \mathbf{P}_{\mathbf{A}}\mathbf{P}_{\mathbf{A}} = \mathbf{P}_{\mathbf{A}}$
  - $\mathbf{P}_{\mathbf{A}} = \mathbf{P}_{\mathbf{A}}^T$ for real $\mathbf{A}$ ($\mathbf{P}_{\mathbf{A}} = \mathbf{P}_{\mathbf{A}}^H$ for complex $\mathbf{A}$)

- additional properties that will be revealed in later lectures:
  - the eigenvalues of $\mathbf{P}_{\mathbf{A}}$ are either zero or one (cf. Eigendecomposition Topic)
  - $\mathbf{P}_{\mathbf{A}}$ can be written as $\mathbf{P}_{\mathbf{A}} = \mathbf{U}_1\mathbf{U}_1^T = \mathbf{P}_{\mathbf{U}_1}$ for some semi-orthogonal $\mathbf{U}_1$ (cf. Singular Value Decomposition Topic)
    - $*$ we can also prove it here:
      - · there always exists a semi-orthogonal $\mathbf{U}_1$ such that $\mathcal{R}(\mathbf{A}) = \mathcal{R}(\mathbf{U}_1)$
      - · $\Pi_{\mathcal{R}(\mathbf{A})}(\mathbf{y}) = \Pi_{\mathcal{R}(\mathbf{U}_1)}(\mathbf{y}) = \mathbf{U}_1\mathbf{U}_1^T\mathbf{y}$
      - · as $\Pi_{\mathcal{R}(\mathbf{A})}(\mathbf{y}) = \Pi_{\mathcal{R}(\mathbf{U}_1)}(\mathbf{y})$ holds for any $\mathbf{y}$, or $(\mathbf{P}_{\mathbf{A}} - \mathbf{U}_1\mathbf{U}_1^T)\mathbf{y} = \mathbf{0}$ for any $\mathbf{y}$, we must have $\mathbf{P}_{\mathbf{A}} = \mathbf{U}_1\mathbf{U}_1^T$
    - $*$ suppose $\mathbf{U}_1 \in \mathbb{R}^{m\times n}$, $\Pi_{\mathcal{R}(\mathbf{U}_1)}(\mathbf{y}) = \mathbf{U}_1\mathbf{U}_1^T\mathbf{y} = \sum_{i=1}^n (\mathbf{u}_{1i}^T\mathbf{y})\mathbf{u}_{1i}$

More on Projections

- **Definition:** a square matrix $\mathbf{P}$ is called a projection matrix (projector) if it is idempotent, i.e, $\mathbf{P}^2 = \mathbf{P}$.
  - easy to understand from a geometric view
  - projection onto $\mathcal{R}(\mathbf{P})$
  - complement projector: $\mathbf{I} - \mathbf{P}$; projection onto $\mathcal{R}(\mathbf{I} - \mathbf{P}) = \mathcal{N}(\mathbf{P})$
  - if $\mathbf{P} \in \mathbb{R}^{m \times m}$, $\mathbb{R}^m = \mathcal{R}(\mathbf{P}) \oplus \mathcal{N}(\mathbf{P})$
- in practice, when we say projection, it mostly refers to orthogonal projection
  - $\mathcal{R}(\mathbf{P})^\perp = \mathcal{N}(\mathbf{P}^T) = \mathcal{N}(\mathbf{P})$ (the complement is the orthogonal complement)
- a projection matrix that is not an orthogonal projection matrix is called an oblique projection matrix



Pseudo-Inverse

- the pseudo-inverse of a full-column-rank $\mathbf{A}$ is defined as

$$\mathbf{A}^\dagger = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T.$$

- $\mathbf{A}^\dagger$ satisfies $\mathbf{A}^\dagger\mathbf{A} = \mathbf{I}$, but not necessarily $\mathbf{A}\mathbf{A}^\dagger = \mathbf{I}$
- $\mathbf{A}^\dagger\mathbf{y}$ is the LS solution
- the orthogonal projector of $\mathbf{A}$ becomes $\mathbf{P}_{\mathbf{A}} = \mathbf{A}\mathbf{A}^\dagger$

- note: a more general definition of the pseudo-inverse for general $\mathbf{A}$ will be studied later (cf. Singular Value Decomposition Topic)

LS and Convex Optimization

- we can also prove the LS optimality condition (Theorem 1) by optimization
- the gradient of a continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is defined as

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

  where $\frac{\partial f}{\partial x_i}$ is the partial derivative
- Fact: consider an unconstrained optimization problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} \; f(\mathbf{x})$$

  where $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable
  - suppose f is convex (we skip the def. here). A point $\mathbf{x}^\star$ is an optimal solution if and only if $\nabla f(\mathbf{x}^\star) = \mathbf{0}$
  - for non-convex f, any point $\hat{\mathbf{x}}$ satisfying $\nabla f(\hat{\mathbf{x}}) = \mathbf{0}$ is a stationary point

LS and Convex Optimization

- Fact: consider a quadratic function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{R}\mathbf{x} + \mathbf{q}^T\mathbf{x} + c,$$

  where $\mathbf{R} \in \mathbb{S}^{n\times n}$.

  - $\nabla f(\mathbf{x}) = \mathbf{R}\mathbf{x} + \mathbf{q}$
  - f is convex if $\mathbf{R}$ is positive semidefinite (PSD); for now it suffices to know that if $\mathbf{R}$ takes the form $\mathbf{R} = \mathbf{A}^T\mathbf{A}$ for some $\mathbf{A}$, it is PSD (easy to verify)

- the LS objective function is

$$f(\mathbf{x}) = \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 = \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} - 2(\mathbf{A}^T\mathbf{y})^T\mathbf{x} + \|\mathbf{y}\|_2^2.$$

  Using the above optimization facts, $\mathbf{x}_{\text{LS}}$ is an LS optimal solution if and only if $\mathbf{A}^T\mathbf{A}\mathbf{x}_{\text{LS}} - \mathbf{A}^T\mathbf{y} = \mathbf{0}$.

- LS problem is one quadratic programming (QP) problem
- the normal equation is equivalent to the first-order optimality (or KKT) condition

LS and Convex Optimization

- using optimization results is handy in some (actually, many) cases
- example (Tikhonov regularization): consider a regularized LS problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2+\lambda\|\mathbf{x}\|_2^2, \qquad \text{for some constant } \lambda>0.$$

  - ℓ_2-norm enforces total smoothness
  - solution by optimization: $\nabla f(\mathbf{x}) = 2\mathbf{A}^T\mathbf{A}\mathbf{x} - 2\mathbf{A}^T\mathbf{y} + 2\lambda\mathbf{x}$. Thus the optimal solution is
$$\mathbf{x}_{\text{RLS}} = (\mathbf{A}^T\mathbf{A}+\lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{y}$$
  - solution by the projection thm., in contrast: have to rewrite the problem as
$$\min_{\mathbf{x}\in\mathbb{R}^n} \left\| \begin{bmatrix}\mathbf{y}\\ \mathbf{0}\end{bmatrix} - \begin{bmatrix}\mathbf{A}\\ \sqrt{\lambda}\mathbf{I}\end{bmatrix}\mathbf{x}\right\|_2^2,$$
    and use the projection theorem to get the same result.
- LS with Tikhonov regularization is commonly used for solving underdeter. linear systems; it can make ill-conditioned (i.e., rank-deficient or close-to-rank-deficient) LS problem to be well-conditioned; LS + Tikhonov reg. = ridge regression model
- if there are $\mathbf{x}$ that satisfy $\mathbf{A}\mathbf{x}=\mathbf{b}$, this will chose the solution with least norm

How to obtain the Solution to a LS?

- direct methods for solving LS
  - method of normal equations
  - QR decomposition
- iterative methods for solving LS
  - gradient descent
  - coordinate descent
  - more...

Direct Methods via Method of Normal Equations

- for LS problem with $\mathbf{A} \in \mathbb{R}^{m \times n}$ of full column rank, the solution is

$$\mathbf{x}_{\mathsf{LS}} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{y}$$

  - naïve computation, complexity: $\mathcal{O}(mn^2 + n^3)$

- For example: solving LS via the normal equations

$$\mathbf{A}^T \mathbf{A} \mathbf{x}_{\mathsf{LS}} = \mathbf{A}^T \mathbf{y}$$

  - solving the above symmetric PD linear system (cf. Linear Systems Topic)
    * compute the lower triangular portion of $\mathbf{C} = \mathbf{A}^T \mathbf{A}$, $\mathcal{O}(mn^2)$
    * form the matrix-vector product $\mathbf{d} = \mathbf{A}^T \mathbf{y}$, $\mathcal{O}(mn)$
    * compute the Cholesky factorization $\mathbf{C} = \mathbf{G}\mathbf{G}^T$, $\mathcal{O}(n^3/3)$
    * solve $\mathbf{G}\mathbf{z} = \mathbf{d}$ and $\mathbf{G}^T \mathbf{x}_{\mathsf{LS}} = \mathbf{z}$, $\mathcal{O}(n^2)$

Direct Methods via QR Decomposition

- LS can be solved by the QR decompositions
- will discuss this in detail in Topic: Orthogonalization and QR Decomposition

Iterative Methods for Solving LS

- in the direct methods for solving LS, we need to solve

$$(\mathbf{A}^T\mathbf{A})\mathbf{x}_{\mathsf{LS}} = \mathbf{A}^T\mathbf{y},$$

  and that requires $\mathcal{O}(n^3)$

  – we also need to compute $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{y}$; their complexities are $\mathcal{O}(mn^2)$ and $\mathcal{O}(mn)$, resp.

- $\mathcal{O}(n^3)$ is expensive for very large n

- Question: can we have cheaper LS solutions, perhaps with some compromise of the solution accuracies?

Gradient Descent

- consider a general unconstrained optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$$

where f is continuously differentiable

- Gradient Descent (GD): given a starting point $\mathbf{x}^{(0)}$, do

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} - \mu\nabla f(\mathbf{x}^{(k-1)}), \quad k = 1, 2, \ldots$$

where $\mu > 0$ is a step size

- take an optimization course to get more details! It is known that
  - for convex f and under some appropriate choice of μ, GD converges to an optimal solution
  - for non-convex f and under some appropriate choice of μ, GD converges to a stationary point

Gradient Descent

- GD for LS:

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} - 2\mu(\mathbf{A}^T\mathbf{A}\mathbf{x}^{(k-1)} - \mathbf{A}^T\mathbf{y}), \quad k = 1, 2, \ldots$$

- complexity for dense $\mathbf{A}$
  - computing $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{y}$: $\mathcal{O}(mn^2)$ and $\mathcal{O}(mn)$, resp. (same as before)
    - $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{y}$ are cached for subsequent use in gradient descent
  - complexity of each iteration: $\mathcal{O}(n^2)$
- complexity for sparse $\mathbf{A}$ (solving (large) sparse LS is an important topic)
  - computing $\mathbf{A}^T\mathbf{y}$: $\mathcal{O}(\mathrm{nnz}(\mathbf{A}))$
  - complexity of each iteration: $\mathcal{O}(n + \mathrm{nnz}(\mathbf{A}))$
    - $\mathbf{A}^T\mathbf{A}$ is not necessarily sparse, so we do $\mathbf{A}\mathbf{x}^{(k-1)}$ and then $\mathbf{A}^T(\mathbf{A}\mathbf{x}^{(k-1)})$

Gradient Descent

- gradient descent is easy to understand, but there are better algorithms...
- further reading: the conjugate gradient method; see, e.g., https://stanford.edu/class/ee364b/lectures/conj_grad_slides.pdf

Online LS

- let $\bar{\mathbf{a}}_i \in \mathbb{R}^n$ denote the ith row of $\mathbf{A}$, then

$$\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 = \sum_{t=1}^{m} |\bar{\mathbf{a}}_t^T \mathbf{x} - y_t|^2$$

- the LS formulation can be written as

$$\min_{\mathbf{x} \in \mathbb{R}^n} \sum_{t=1}^{m} |\bar{\mathbf{a}}_t^T \mathbf{x} - y_t|^2$$

- the LS we learnt is a batch process; i.e., solve one $\mathbf{x}$ given the whole $(\mathbf{A}, \mathbf{y})$; the afore-mentioned GD method is also hence referred to as batch GD
- there are many applications where new $(\bar{\mathbf{a}}_t, y_t)$ appears as time goes, and we want the process to be adaptive or in real time; i.e., $\mathbf{x}$ is updated with t
- alternatively, we want something cheaper than gradient descent

Incremental Gradient Descent

- consider an optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{t=1}^{m} f_t(\mathbf{x})$$

  where every f_t is continuously differentiable

- Incremental Gradient Descent:

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \mu \nabla f_t(\mathbf{x}_{t-1}), \quad t = 1, 2, \ldots$$

  – also called online gradient descent, stochastic gradient descent (SGD), least mean squares (LMS) (in 70's), ...

- incremental gradient descent for LS:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + 2\mu(y_t - \bar{\mathbf{a}}_t^T \mathbf{x}_{t-1})\bar{\mathbf{a}}_t$$

- complexity: $\mathcal{O}(n)$
  – commonly used in large-scale optimization like learning neural networks

Recursive LS

- Recursive LS (RLS) formulation:

$$\mathbf{x}_t = \arg\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{i=1}^{t} \lambda^{t-i} |\bar{\mathbf{a}}_i^T \mathbf{x} - y_i|^2$$

  where $0 < \lambda \leq 1$ is a prescribed constant and is called the forgetting or discounting factor

  - weigh the importance of $|\bar{\mathbf{a}}_i^T \mathbf{x} - y_i|^2$ w.r.t. time t; the present is most important; distant pasts are insignificant; how much we remember the pasts depends on λ

- at first look, the RLS solution is $\mathbf{x}_t = \mathbf{R}_t^{-1}\mathbf{q}_t$, where

$$\mathbf{R}_t = \sum_{i=1}^{t} \lambda^{t-i} \bar{\mathbf{a}}_i \bar{\mathbf{a}}_i^T, \quad \mathbf{q}_t = \sum_{i=1}^{t} \lambda^{t-i} y_i \bar{\mathbf{a}}_i$$

- a recursive formula for $\mathbf{x}_t$ can be derived by using the Woodbury matrix identity and by using the problem structures carefully

Woodbury Matrix Identity

For $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ of appropriate dimensions, we have

$$(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1} = \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1},$$

assuming that the inverses above exist.

- for the RLS problem, it is sufficient to know the special case

$$(\mathbf{A} + \mathbf{b}\mathbf{b}^T)^{-1} = \mathbf{A}^{-1} - \frac{1}{1 + \mathbf{b}^T\mathbf{A}^{-1}\mathbf{b}}\mathbf{A}^{-1}\mathbf{b}\mathbf{b}^T\mathbf{A}^{-1}$$

Recursive LS

- it can be verified that $\mathbf{R}_t = \lambda \mathbf{R}_{t-1} + \bar{\mathbf{a}}_t \bar{\mathbf{a}}_t^T$, $\mathbf{q}_t = \lambda \mathbf{q}_{t-1} + y_t \bar{\mathbf{a}}_t$
- by the Woodbury matrix identity,

$$\mathbf{R}_t^{-1} = (\lambda \mathbf{R}_{t-1} + \bar{\mathbf{a}}_t \bar{\mathbf{a}}_t^T)^{-1} = \tfrac{1}{\lambda}\mathbf{R}_{t-1}^{-1} - \frac{1}{1 + \frac{1}{\lambda}\bar{\mathbf{a}}_t^T \mathbf{R}_{t-1}^{-1} \bar{\mathbf{a}}_t} (\tfrac{1}{\lambda}\mathbf{R}_{t-1}^{-1}\bar{\mathbf{a}}_t)(\tfrac{1}{\lambda}\mathbf{R}_{t-1}^{-1}\bar{\mathbf{a}}_t)^T$$

- let $\mathbf{P}_t = \mathbf{R}_t^{-1}$ and $\mathbf{g}_t = \dfrac{1}{1 + \frac{1}{\lambda}\bar{\mathbf{a}}_t^T \mathbf{R}_{t-1}^{-1} \bar{\mathbf{a}}_t} (\tfrac{1}{\lambda}\mathbf{R}_{t-1}^{-1}\bar{\mathbf{a}}_t)$. We have

$$\begin{aligned}
\mathbf{g}_t &= \frac{1}{1 + \frac{1}{\lambda}\bar{\mathbf{a}}_t^T \mathbf{P}_{t-1} \bar{\mathbf{a}}_t} (\tfrac{1}{\lambda}\mathbf{P}_{t-1}\bar{\mathbf{a}}_t) \\
\mathbf{P}_t &= \tfrac{1}{\lambda}\mathbf{P}_{t-1} - \mathbf{g}_t (\tfrac{1}{\lambda}\mathbf{P}_{t-1}\bar{\mathbf{a}}_t)^T \\
\mathbf{x}_t = \mathbf{P}_t \mathbf{q}_t &= \mathbf{P}_{t-1}\mathbf{q}_{t-1} - \lambda \mathbf{g}_t (\tfrac{1}{\lambda}\mathbf{P}_{t-1}\bar{\mathbf{a}}_t)^T \mathbf{q}_{t-1} + \tfrac{1}{\lambda} y_t \mathbf{P}_{t-1}\bar{\mathbf{a}}_t - y_t \mathbf{g}_t (\tfrac{1}{\lambda}\mathbf{P}_{t-1}\bar{\mathbf{a}}_t)^T \bar{\mathbf{a}}_t \\
&= \mathbf{x}_{t-1} - (\bar{\mathbf{a}}_t^T \mathbf{x}_{t-1})\mathbf{g}_t + y_t \mathbf{g}_t \\
&= \mathbf{x}_{t-1} + (y_t - \bar{\mathbf{a}}_t^T \mathbf{x}_{t-1})\mathbf{g}_t
\end{aligned}$$

Recursive LS

- summary of the RLS recursion:

$$\mathbf{g}_t = \frac{1}{1 + \frac{1}{\lambda}\bar{\mathbf{a}}_t^T \mathbf{P}_{t-1}\bar{\mathbf{a}}_t}(\tfrac{1}{\lambda}\mathbf{P}_{t-1}\bar{\mathbf{a}}_t)$$
$$\mathbf{P}_t = \tfrac{1}{\lambda}\mathbf{P}_{t-1} - \mathbf{g}_t(\tfrac{1}{\lambda}\mathbf{P}_{t-1}\bar{\mathbf{a}}_t)^T$$
$$\mathbf{x}_t = \mathbf{x}_{t-1} + (y_t - \bar{\mathbf{a}}_t^T \mathbf{x}_{t-1})\mathbf{g}_t$$

- complexity: $\mathcal{O}(n)$
- remarks:
  - comparison with incremental gradient descent: it replaces $\mathbf{g}_t$ with $2\mu\bar{\mathbf{a}}_t$
  - the above RLS recursion may be numerically unstable as empirical results suggested (further reading: **[Liavas-Regalia'99]**); modified RLS schemes were developed to mend this issue

Coordinate Descent

- the problem is to solve

$$\min_{\mathbf{x}\in\mathbb{R}^n} \ f(\mathbf{x})$$

input: a starting point $\mathbf{x}^{(0)}$
for $k = 0, 1, 2, \ldots$

$$x_1^{(k+1)} = \arg\min_{x_1\in\mathbb{R}} f(x_1, x_2^{(k)}, \ldots, x_n^{(k)})$$
$$x_2^{(k+1)} = \arg\min_{x_2\in\mathbb{R}} f(x_1^{(k+1)}, x_2, \ldots, x_n^{(k)})$$
$$\vdots$$
$$x_n^{(k+1)} = \arg\min_{x_n\in\mathbb{R}} f(x_1^{(k+1)}, x_2^{(k+1)}, \ldots, x_n)$$

end

- a.k.a. nonlinear Gauss-Seidel
- It is known that **[Tseng'01]**
  - Convergence guarantees toward a local optimal (minimal) point for smooth functions or separable functions
  - No convergence toward a minimum for non-separable and non-smooth functions: some points (coordinatewise minimum) get stuck

Coordinate Descent

- CD for LS. notice

$$f(\mathbf{x}) = \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 = \sum_{i=1}^{m} \|\mathbf{a}_i x_i - \mathbf{y}\|_2^2 = f(x_1, \ldots, x_n)$$

and

$$\nabla f(\mathbf{x}) = 2\mathbf{A}^T(\mathbf{A}\mathbf{x} - \mathbf{y}) = 2 \begin{bmatrix} \mathbf{a}_1^T(\mathbf{A}\mathbf{x} - \mathbf{y}) \\ \vdots \\ \mathbf{a}_n^T(\mathbf{A}\mathbf{x} - \mathbf{y}) \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

- minimize w.r.t x_i for $i = 1, \ldots, n$, with fixed x_j $(j \neq i)$

$$0 = \frac{\partial f}{\partial x_i} = \mathbf{a}_i^T(\mathbf{A}\mathbf{x} - \mathbf{y}) = \mathbf{a}_i^T(\mathbf{a}_i x_i + \sum_{j \neq i} \mathbf{a}_j x_j - \mathbf{y})$$

we have

$$x_i = \frac{\mathbf{a}_i^T(\mathbf{y} - \sum_{j \neq i} \mathbf{a}_j x_j)}{\|\mathbf{a}_i\|_2^2}$$

Coordinate Descent

input: a starting point $\mathbf{x}^{(0)}$
for $k = 0, 1, 2, \ldots$
for $i = 1, 2, \ldots, n$

$$x_i^{(k+1)} = \frac{\mathbf{a}_i^T(\mathbf{y} - \sum_{j=1}^{i-1} \mathbf{a}_j x_j^{(k+1)} - \sum_{j=i+1}^{n} \mathbf{a}_j x_j^{(k)})}{\|\mathbf{a}_i\|_2^2}$$

end
end

- equivalent to solving linear system $\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{y}$ via CD (cf. Linear Sys. Topic)
- clever update scheme can be developed with low memory impact
- CD is a non-gradient optimization (or a derivative-free optimization) method; it can be extremely fast
- possibly visit the coordinate in arbitrary order (cycle, random, more refined methods,etc.)
- Block coordinate descent (BCD): update not only one coordinate, but a bunch of them, i.e., optimizing according to problem architecture; a.k.a. block nonlinear Gauss-Seidel

More Iterative Methods for LS

- accelerated GD
- conjugate GD
- Newton's method
- Gauss-Newton method (line search)
- the Levenberg-Marquardt method (trust region)
- ...

Beyond LS

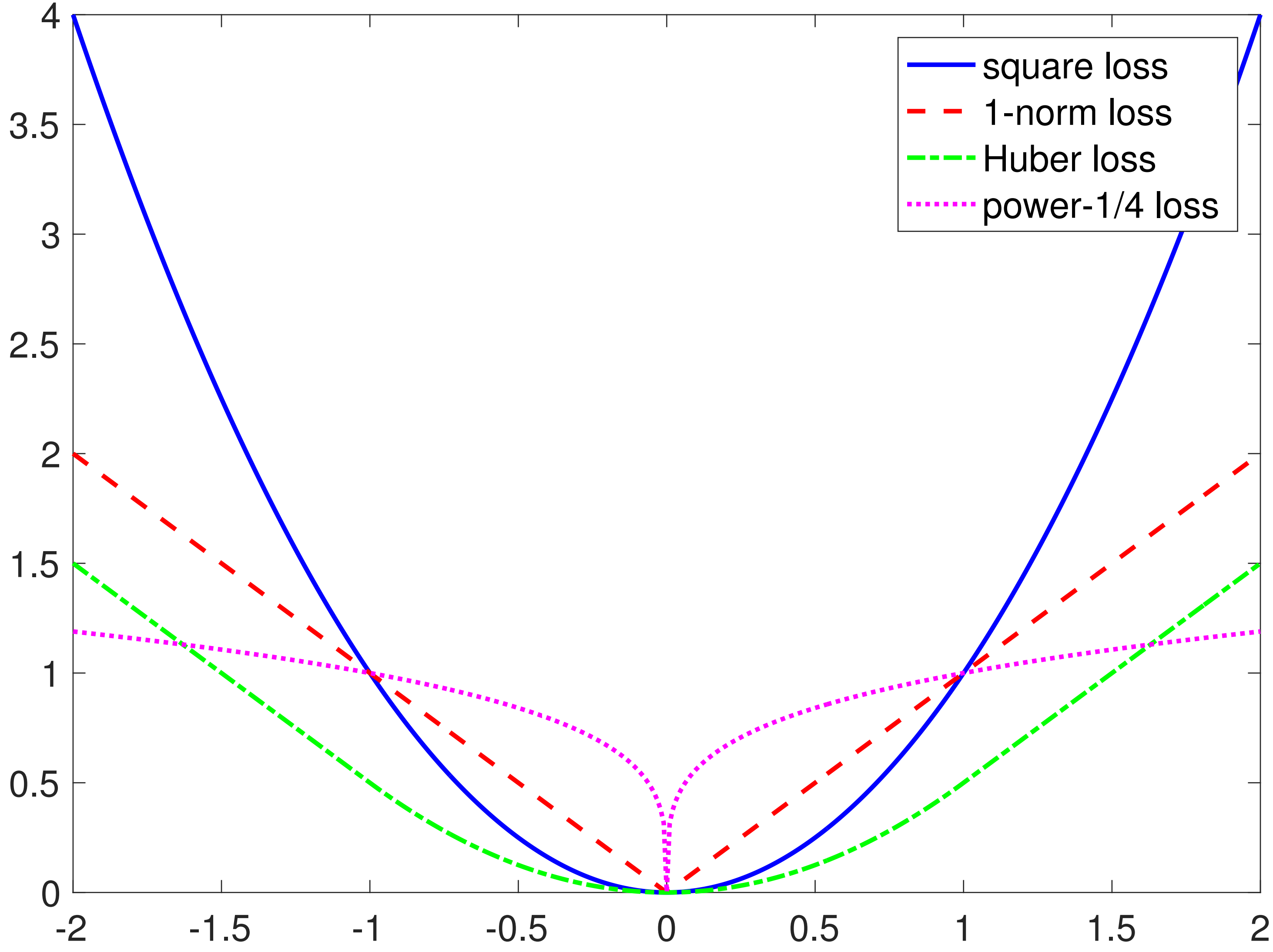
- The LS problem can be represented as

$$\min_{\mathbf{x}\in\mathbb{R}^n} \sum_{i=1}^{m} \ell(\bar{\mathbf{a}}_i^T \mathbf{x} - y_i)$$

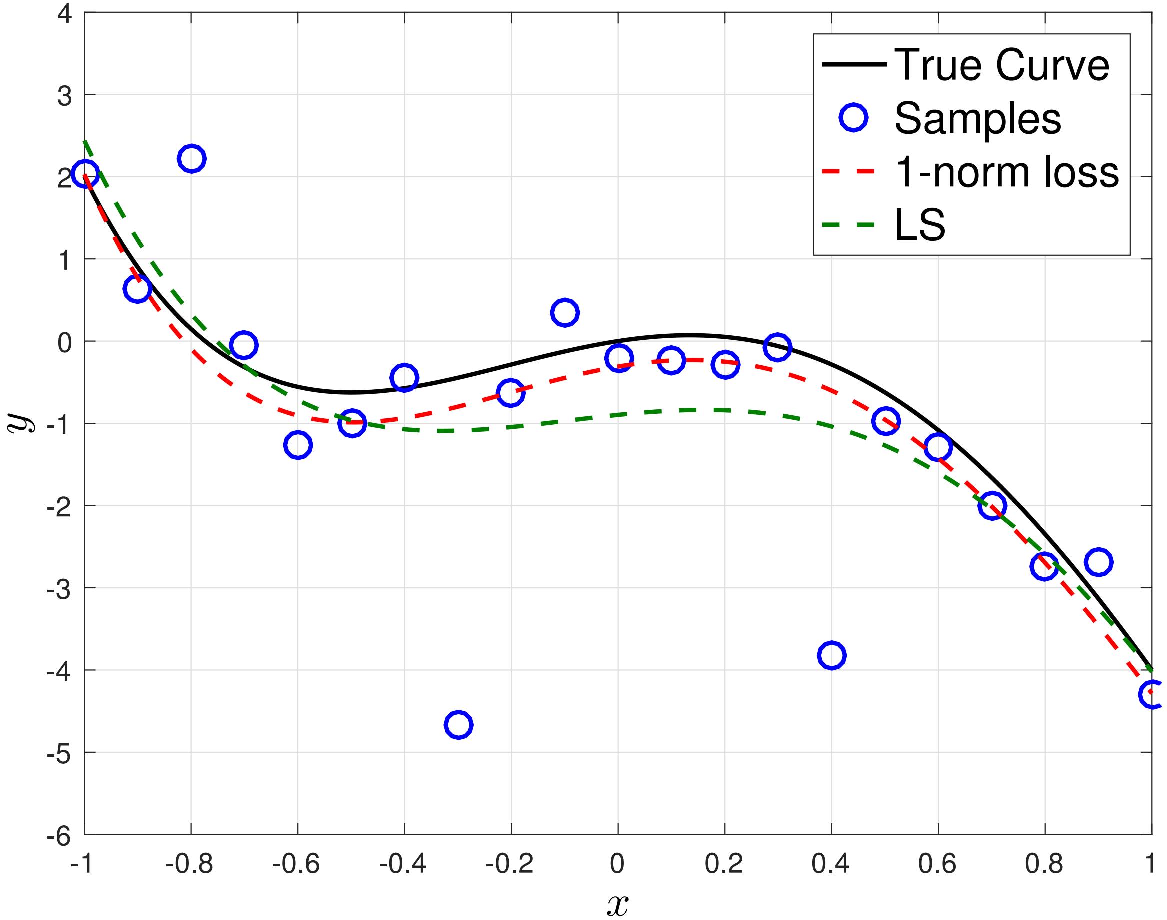
  where $\ell(z) = |z|^2$ denotes the loss function for measuring the badness of fit

- Question: why don't we use other loss functions?
  - we can indeed use other loss functions, such as
    - 1-norm loss: $\ell(z) = |z|$ (least absolute deviations (LAD))
    - Huber loss: $\ell(z) = \begin{cases} \frac{1}{2}|z|^2, & |z| \leq 1 \\ |z| - \frac{1}{2}, & |z| > 1 \end{cases}$
    - power-p loss: $\ell(z) = |z|^p$, with $p < 1$
  - corresponding to different prior distributions for noise $\mathbf{v}$ in $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{v}$
  - the above loss functions are more robust against outliers, but
  - they require optimization and don't result in a clean closed-form solution as LS (a method to solve them is iteratively reweighted least squares (IRLS); in each iteration, a (weighted) LS is solved via successive linear approximation (SLA))

Illustration of Loss Functions



Curve Fitting Example



"True" curve: the true $f(x)$, $p = 5$. The points at $x = -0.3$ and $x = 0.4$ are outliers, and they do not follow the true curve. The 1-norm loss problem is solved by a convex optimization tool.

More on LS

more topics related to LS in future lectures (cf. LS Revisited and Sparse Opt. Topic)

- linear LS (ordinary, weighted, generalized...) vs. nonlinear LS (neural networks...)
- regularized LS
  - penalized LS (e.g., ℓ_0/best subset, ℓ_1/lasso, ℓ_2/ridge, ℓ_1+ℓ_2/elastic net, ...)
  - constrained LS (e.g., non-negative LS, bounded-variable LS, linearly constrained LS, LS with simplex constraints, LS with norm ball constraint, ...)
- underdetermined linear system of equations
  - find the minimum ℓ_2 solution of an underdetermined linear system
  - find the minimum ℓ_0 solution of an underdetermined linear system
  - find the minimum ℓ_1 solution of an underdetermined linear system
  - majorization-minimization for ℓ_2–ℓ_1 minimization
  - dictionary learning and frame learning
- LS with errors in $\mathbf{A}$
  - total LS
  - robust LS, and its equivalence to regularized LS

Part III: Matrix Factorization

Matrix Factorization

There are also many applications in which we deal with a representation of multiple given $\mathbf{y}_i$'s via

$$\mathbf{y}_i = \mathbf{A}\mathbf{b}_i + \mathbf{v}_i, \quad i = 1, \ldots, n,$$

where $\mathbf{A} \in \mathbb{R}^{n \times k}$, $\mathbf{b}_i \in \mathbb{R}^k$, $i = 1, \ldots, n$; $\mathbf{v}_i$'s are noise. In particular, both $\mathbf{b}_i$'s and $\mathbf{A}$ are to be determined.

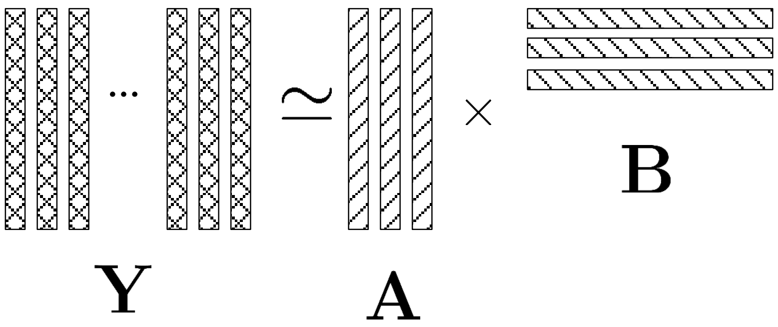
- for example, in basis representation, we want to jointly learn the dictionary from data

Matrix Factorization

Problem: given $\mathbf{Y} \in \mathbb{R}^{m \times n}$ and a positive integer $k < \min\{m, n\}$, solve

$$\min_{\mathbf{A}\in\mathbb{R}^{m\times k},\mathbf{B}\in\mathbb{R}^{k\times n}} \|\mathbf{Y} - \mathbf{AB}\|_F^2$$

where $\|\mathbf{Y} - \mathbf{AB}\|_F^2 = \sum_{i=1}^n \|\mathbf{y}_i - \mathbf{A}\mathbf{b}_i\|_2^2 = \sum_{i=1}^m \|\bar{\mathbf{y}}_i - \mathbf{B}^T\bar{\mathbf{a}}_i\|_2^2 = \sum_{i,j} |y_{ij} - [\mathbf{AB}]_{ij}|^2$ with $\bar{\mathbf{y}}_i \in \mathbb{R}^n, \bar{\mathbf{a}}_i \in \mathbb{R}^k$ denoting the ith row of $\mathbf{Y}, \mathbf{A}$, respectively.



- matrix factorization (MF) is also called low-rank matrix factorization or low-rank matrix approximation: let $\mathbf{Z} = \mathbf{AB}$. It has $\mathrm{rank}(\mathbf{Z}) \leq k$.
- like in LS, we may often want to add constraints and/or penalties in MF problems, like orthogonality constraint, non-negative constraint, linear constraint, sparsity constraint, etc.

Principal Component Analysis

Aim: given a collection of data points $\mathbf{y}_1, \ldots, \mathbf{y}_n \in \mathbb{R}^m$, perform a low-dimensional representation

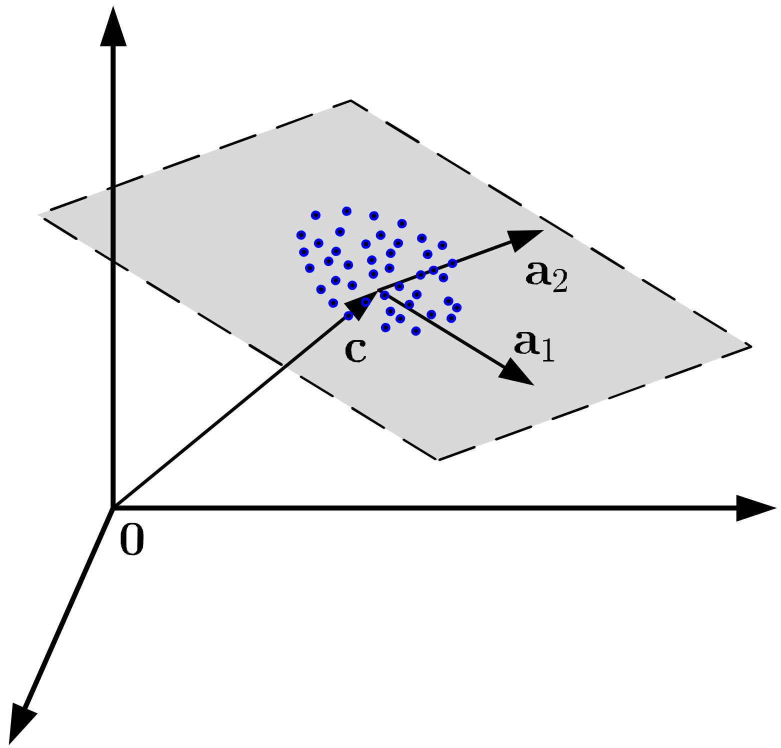
$$\mathbf{y}_i = \mathbf{A}\mathbf{b}_i + \mathbf{c} + \mathbf{v}_i, \quad i = 1, \ldots, n,$$

where $\mathbf{A} \in \mathbb{R}^{m \times k}$ is a basis matrix; $\mathbf{b}_i \in \mathbb{R}^k$ is the coefficient for $\mathbf{y}_i$; $\mathbf{c} \in \mathbb{R}^m$ is the base or mean in statistics terms; $\mathbf{v}_i$ is noise or modeling error.

- Principal component analysis (PCA):
  - choose $\mathbf{c} = \frac{1}{n}\sum_{i=1}^n \mathbf{y}_i$
  - let $\bar{\mathbf{y}}_i = \mathbf{y}_i - \mathbf{c}$, and solve

$$\min_{\mathbf{A},\mathbf{B}} \ \|\bar{\mathbf{Y}} - \mathbf{A}\mathbf{B}\|_F^2$$

  - we may also want a semi-orthogonal $\mathbf{A}$
- in PCA problem, minimizing squared distances equals maximizing variance



Principal Component Analysis

- applications: dimensionality reduction, visualization of high-dimensional data, compression, extraction of meaningful features from data,...

- an example:
  - senate voting: `http://livebooklabs.com/keeppies/c5a5868ce26b8125`

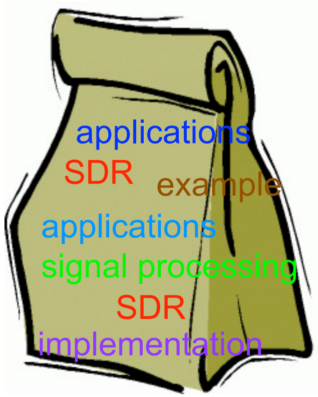
Topic Modeling

Aim: discover thematic information, or topics, from a (often large) collection of documents, such as books, articles, news, blogs,...

- bag-of-words representation: represent each document as a vector of word counts

… In fact, we will soon see that the implementation of SDR can be very easy, which allows signal processing practitioners to quickly test the viability of SDR in their applications. Several highly successful applications will be showcased as examples ……

a document



bag of words

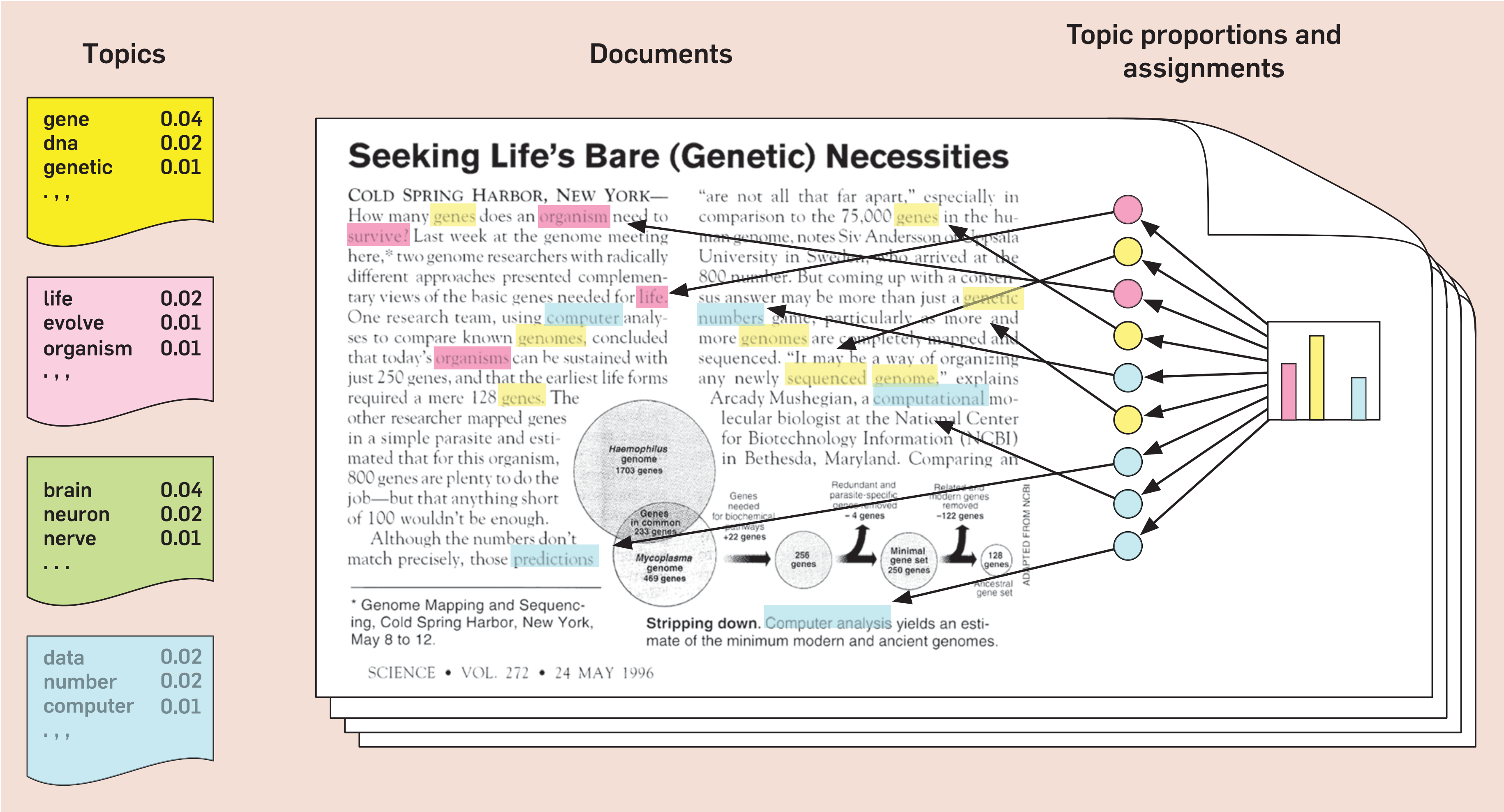
$$
\mathbf{y} = \begin{array}{cl} \textbf{count} & \textbf{term} \\ \left[\begin{array}{c} 0 \\ 2 \\ 2 \\ 0 \\ 1 \\ 1 \\ \vdots \\ 1 \end{array}\right] & \begin{array}{l} \text{efficiency} \\ \text{applications} \\ \text{SDR} \\ \text{communications} \\ \text{example} \\ \text{signal processing} \\ \vdots \\ \text{implementation} \end{array} \end{array}
$$

bag–of–words representation

Topic Modeling

- let n be the number of documents
- let $\mathbf{y}_i \in \mathbb{R}^m$ be the bag-of-words representation of the ith document, $i = 1, \ldots, n$
- let $\mathbf{Y} = [\ \mathbf{y}_1, \ldots \mathbf{y}_n\ ] \in \mathbb{R}^{m \times n}$, called the term-document matrix
- hypotheses: **[Turney-Pantel'10]**
  - if documents have similar columns vectors in $\mathbf{Y}$, or similar usage of words, they tend to have similar meanings
  - the topic of a document will probabilistically influence the author's choice of words when writing the document

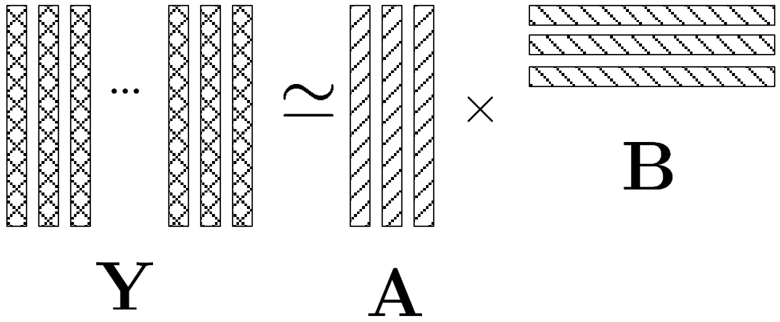
Topic Modeling



Source: [Blei'12].

Topic Modeling

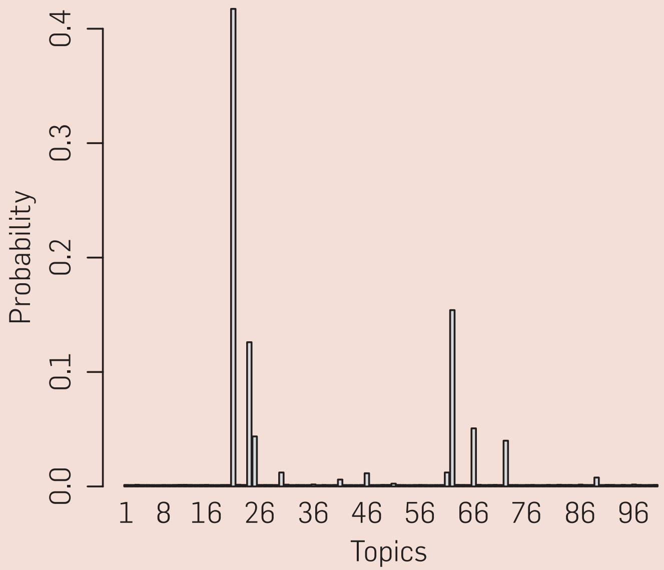
- Problem: apply matrix factorization to a term-document matrix $\mathbf{Y}$



  - $\mathbf{A}$ is called a term-topic matrix, $\mathbf{B}$ is called a topic-document matrix

- Interpretation:
  - each column $\mathbf{a}_i$ of $\mathbf{A}$ should represent a theme topic, e.g., local affairs, foreign affairs, politics, sports... in a collection of newspapers
  - as $\mathbf{y}_i \approx \mathbf{A}\mathbf{b}_i$, each document is postulated as a linear combination of topics
  - matrix factorization aims at discovering topics from the documents

Topic Modeling



“Genetics”	“Evolution”	“Disease”	“Computers”
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

Topics found in a real set of documents. Source: **[Blei’12]**. The document set consists of $17,000$ articles from the journal *Science*. The topics are discovered using a technique called *latent Dirichlet allocation*, which is not the same as, but has strong connections to, matrix factorization.

Topic Modeling

- topic modeling via matrix factorization has been used in, or is tightly connected to
  - information retrieval, natural language processing, machine learning
  - document clustering, classification and retrieval
  - latent semantic analysis, latent semantic indexing: finding similarities of documents, finding similarities of terms (are "cars," "Lamborghini," and "Ferrari" related?)
- though not considered in this course, it seems better to also model $\mathbf{A}$, $\mathbf{B}$ as element-wise non-negative—this will lead to *non-negative matrix factorization*
- further reading: **[Turney-Pantel'10]**
  - as an aside, it mentions a related application where computers can achieve a score of 92.5% on multiple-choice synonym questions from TOEFL, whereas the average human score is 64.5%

Matrix Factorization

The matrix factorization problem

$$\min_{\mathbf{A}\in\mathbb{R}^{m\times k},\mathbf{B}\in\mathbb{R}^{k\times n}} \|\mathbf{Y}-\mathbf{A}\mathbf{B}\|_F^2$$

- has non-unique factors
  - suppose $(\mathbf{A}^\star, \mathbf{B}^\star)$ is an optimal solution to the problem, and let $\mathbf{Q} \in \mathbb{R}^{k\times k}$ be any nonsingular matrix. Then $(\mathbf{A}^\star\mathbf{Q}^{-1}, \mathbf{Q}\mathbf{B}^\star)$ is also an optimal solution.
  - the non-uniqueness of $(\mathbf{A}, \mathbf{B})$ makes the above matrix factorization formulation a bad formulation for problems such as topic modeling

- is non-convex, but can be solved by singular value decomposition (beautifully) (cf. Singular Value Decomposition Topic)

- can also be handled by LS

Matrix Factorization

- Alternating LS (ALS): given a starting point $(\mathbf{A}^{(0)}, \mathbf{B}^{(0)})$, do

$$\mathbf{A}^{(i+1)} = \arg\min_{\mathbf{A}\in\mathbb{R}^{m\times k}} \|\mathbf{Y} - \mathbf{A}\mathbf{B}^{(i)}\|_F^2$$

$$\mathbf{B}^{(i+1)} = \arg\min_{\mathbf{B}\in\mathbb{R}^{k\times n}} \|\mathbf{Y} - \mathbf{A}^{(i+1)}\mathbf{B}\|_F^2$$

  for $i = 0, 1, 2, \ldots$, and stop when a stopping rule is satisfied.
- let's make a mild assumption that $\mathbf{A}^{(i)}, \mathbf{B}^{(i)}$ have full rank at every i. Then,
  $\mathbf{A}^{(i+1)} = \mathbf{Y}(\mathbf{B}^{(i)})^T(\mathbf{B}^{(i)}(\mathbf{B}^{(i)})^T)^{-1}, \quad \mathbf{B}^{(i+1)} = ((\mathbf{A}^{(i+1)})^T\mathbf{A}^{(i+1)})^{-1}(\mathbf{A}^{(i+1)})^T\mathbf{Y}$
- a special case of alternating minimization (AM, AltMin) or BCD
- ALS is guaranteed to converge an optimal solution to $\min_{\mathbf{A},\mathbf{B}} \|\mathbf{Y} - \mathbf{A}\mathbf{B}\|_F^2$ under some mild assumptions **[Udell-Horn-Zadeh-Boyd'16]**
  - note: this result is specific and does not directly carry forward to other related problems such as low-rank matrix completion
- you can also apply GD, SGD, alternating GD, etc.

Low-Rank Matrix Completion

- let $\mathbf{Y} \in \mathbb{R}^{m \times n}$ be a matrix with missing entries, i.e., the values y_{ij}'s are known only for $(i, j) \in \Omega$ where Ω is an index set that indicates the available entries
- Aim: recover the missing entries of $\mathbf{Y}$
- application: recommender system, data science
- example: movie recommendation (further reading: **[Koren-Bell-Volinsky'09]**)
  - $\mathbf{Y}$ records how user i likes movie j
  - $\mathbf{Y}$ has lots of missing entries; a user doesn't watch all movies
  - $\mathbf{Y}$ may be assumed to have low rank; research shows that only a few factors affect users' preferences.

$$\begin{array}{cc} & \text{movies} \\ \mathbf{Y} = & \begin{bmatrix} 2 & 3 & 1 & ? & ? & 5 & 5 \\ 1 & ? & 4 & 2 & ? & ? & ? \\ ? & 3 & 1 & ? & 2 & 2 & 2 \\ ? & ? & ? & 3 & ? & 1 & 5 \end{bmatrix} & \text{users} \end{array}$$

Low-Rank Matrix Completion

- Problem: given $\{y_{ij}\}_{(i,j)\in\Omega}$, Ω and a positive integer k, solve

$$\min_{\mathbf{A}\in\mathbb{R}^{m\times k},\mathbf{B}\in\mathbb{R}^{k\times n}} \sum_{(i,j)\in\Omega} |y_{ij} - [\mathbf{AB}]_{ij}|^2$$

- ALS can be applied; more tedious to write out the LS solutions than the previous matrix factorization problem but not any harder in principle

- supposingly a very difficult problem, but

- methods like ALS were found to work by means of empirical studies

- recent theoretical research suggests that matrix completion may not be that hard under some assumptions, e.g., ALS can give good results **[Sun-Luo'16]**

Low-Rank Matrix Completion

- an ALS alternative to matrix completion (easier to program):
  - consider an equivalent reformulation of the matrix completion problem

$$\min_{\mathbf{A}\in\mathbb{R}^{m\times k},\mathbf{B}\in\mathbb{R}^{k\times n},\mathbf{R}\in\mathbb{R}^{m\times n}} \|\mathbf{Y}+\mathbf{R}-\mathbf{A}\mathbf{B}\|_F^2 \quad \text{s.t. } r_{ij}=0,\ (i,j)\in\Omega$$

  - do alternating optimization

$$\mathbf{A}^{(i+1)} = \arg\min_{\mathbf{A}\in\mathbb{R}^{m\times k}} \|\mathbf{Y}-\mathbf{A}\mathbf{B}^{(i)}+\mathbf{R}^{(i)}\|_F^2$$

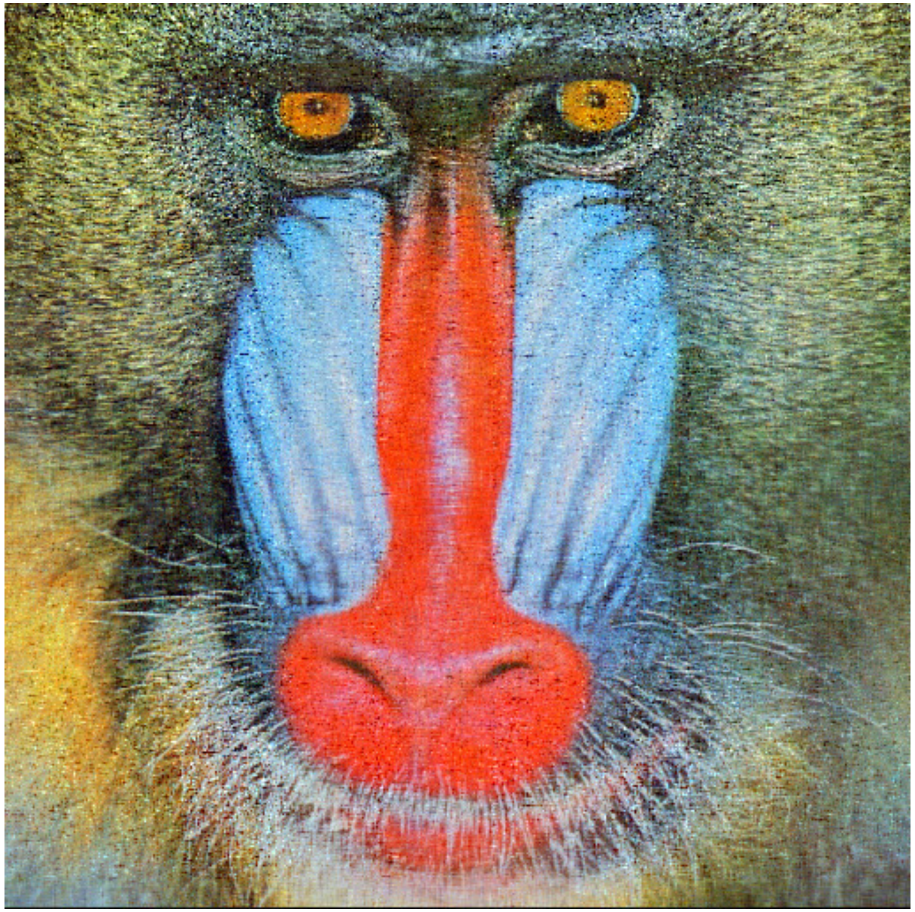
$$\mathbf{B}^{(i+1)} = \arg\min_{\mathbf{B}\in\mathbb{R}^{k\times n}} \|\mathbf{Y}-\mathbf{A}^{(i+1)}\mathbf{B}+\mathbf{R}^{(i)}\|_F^2$$

$$\mathbf{R}^{(i+1)} = \arg\min_{\mathbf{R}\in\mathbb{R}^{m\times n}} \|\mathbf{Y}-\mathbf{A}^{(i+1)}\mathbf{B}^{(i+1)}+\mathbf{R}\|_F^2$$

  the first two are LS as before; the third has a closed form

$$r_{ij}^{(i+1)} = \begin{cases} 0, & (i,j)\in\Omega \\ -[\mathbf{Y}-\mathbf{A}^{(i+1)}\mathbf{B}^{(i+1)}]_{i,j}, & (i,j)\notin\Omega \end{cases}$$

Toy Demonstration of Low-Rank Matrix Completion



Left: An incomplete image with 40% missing pixels. Right: the matrix completion result of the algorithm shown on last page. $k = 120$.

References

[Stoica-Moses'97] P. Stoica and R. L. Moses, *Introduction to Spectral Analysis*, Prentice Hall, 1997.

[Aharon-Elad-Bruckstein'06] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Image Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[Golub-Van Loan'13] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th edition, JHU Press, 2013.

[Sayed-Tarighat-Khajehnouri'05] A. H. Sayed, A. Tarighat, and N. Khajehnouri. "Network-based wireless location," *IEEE Signal Process. Mag.*, vol. 22, no. 4, pp. 24–40, 2005.

[Tseng'01] P. Tseng. "Convergence of a block coordinate descent method for nondifferentiable minimization," *J. Optim. Theory Appl.*, vol. 109, no. 3, pp. 475?494, 2001.

[Turney-Pantel'10] P. D. Turney and P. Pantel, "From frequency to meaning: Vector space models of semantics," *Journal of Artificial Intelligence Research*, vol. 37, pp. 141–188, 2010.

[Blei'12] D. Blei, "Probabilistic topic models," *Communications of the ACM*, vol. 55, no. 4, pp. 77–84, 2012.

[Udell-Horn-Zadeh-Boyd'16] M. Udell, C. Horn, R. Zadeh, and S. Boyd, "Generalized low rank models", *Foundations and Trends in Machine Learning*, 2016.

[Koren-Bell-Volinsky'09] B. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Computer*, vol. 42 no. 8, pp. 30–37, 2009.

[Sun-Luo'16] R. Sun and Z.-Q. Luo, "Guaranteed matrix completion via non-convex factorization." *IEEE Trans. Inform. Theory,* vol. 62, no. 11, pp. 6535–6579, 2016.

[Liavas-Regalia'99] A. P. Liavas and P. A. Regalia, "On the numerical stability and accuracy of the conventional recursive least squares algorithm." *IEEE Trans. Signal Process.,* vol. 47, no. 1, pp. 88-96, 1999.