Matrix Computations
Chapter 3: Least-squares Problems and QR Decomposition

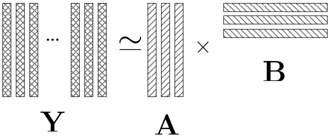
Section 3.4 Problems Related to Least Squares

Jie Lu
ShanghaiTech University

Matrix Factorization

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

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



Also called low-rank matrix approximation

- $\operatorname{rank}(\mathbf{AB}) \le k$

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

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

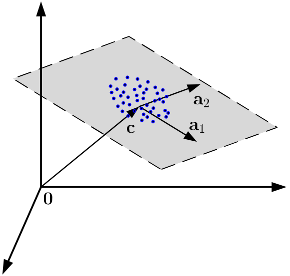
where $\mathbf{A} \in \mathbb{R}^{m \times k}$ is a basis matrix, $\mathbf{b}_i \in \mathbb{R}^k$ is the coefficient for $\mathbf{y}_i$, $\mathbf{c} \in \mathbb{R}^m$ is the base or mean in statistics terms, and $\mathbf{v}_i$ is noise or modeling error

Principal component analysis (PCA):

1. Choose $\mathbf{c} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{y}_i$
2. Let $\bar{\mathbf{y}}_i = \mathbf{y}_i - \mathbf{c}$, and solve

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

3. we may want a semi-orthogonal $\mathbf{A}$



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

- Example of senate voting: `http://livebooklabs.com/keeppies/c5a5868ce26b8125`

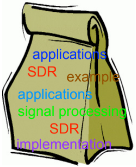
Topic Modeling

Aim: Discover thematic information or topics from a large collection of documents (e.g., 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

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

bag–of–words representation

Topic Modeling (cont'd)

- Let n be the number of documents

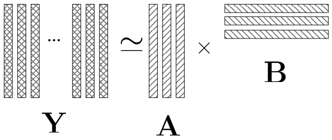
- Let $\mathbf{y}_i \in \mathbb{R}^m$ be the bag-of-words representation of the ith document
- $\mathbf{Y} = [\ \mathbf{y}_1, \ldots \mathbf{y}_n\ ] \in \mathbb{R}^{m \times n}$ is called the term-document matrix

- Hypotheses:[1]
  - 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

[1] 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.

Topic Modeling (cont'd)

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



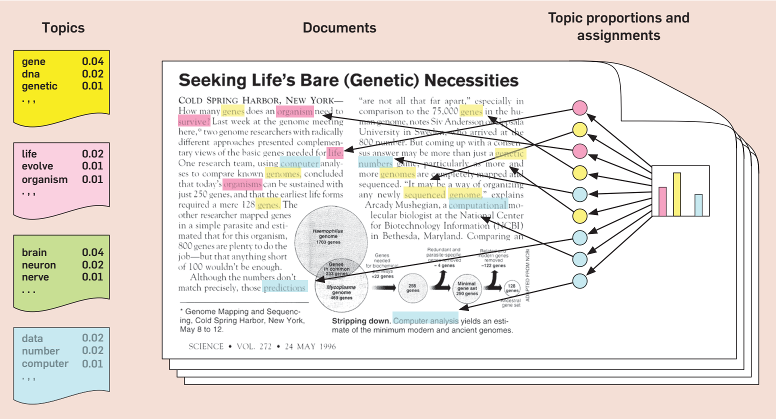
$\mathbf{A}$ is called a term-topic matrix and $\mathbf{B}$ is called a topic-document matrix

Interpretation:

- Each column $\mathbf{a}_i$ of $\mathbf{A}$ represents a theme topic (e.g., local affairs, foreign affairs, politics, sports)
- $\mathbf{y}_i \approx \mathbf{A}\mathbf{b}_i$: each document is postulated as a linear combination of topics
- Matrix factorization aims at discovering topics from the documents

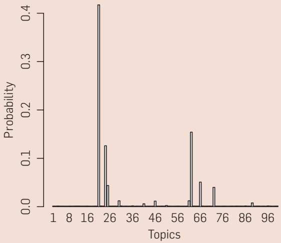
Topic modeling via matrix factorization has been used in or is tightly connected to information retrieval, natural language processing, machine learning; document clustering, classification and retrieval; latent semantic analysis, latent semantic indexing: finding similarities of documents, similarities of terms, etc.

Topic Modeling (cont'd)



Source: D. Blei, "Probabilistic topic models," *Communications of the ACM*, vol. 55, no. 4, pp. 77–84, 2012.

Topic Modeling (cont'd)



"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. 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 [Blei'12]

Matrix Factorization

Problem:

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

The problem has non-unique solutions

- If $(\mathbf{A}^\star, \mathbf{B}^\star)$ is an optimal solution to the problem, then $(\mathbf{A}^\star\mathbf{Q}^{-1}, \mathbf{Q}\mathbf{B}^\star)$ is also an optimal solution for any nonsingular $\mathbf{Q} \in \mathbb{R}^{k\times k}$
- The non-uniqueness of solution makes it a bad formulation for problems such as topic modeling

The problem is non-convex, but can be solved by singular value decomposition (beautifully)

It can also be solved by LS approach

Alternating LS for 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 termination criterion is satisfied

Make a mild assumption that $\mathbf{A}^{(i)}, \mathbf{B}^{(i)}$ have full rank at every i

Alternating LS for Matrix Factorization (cont'd)

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

Alternating LS for Matrix Factorization (cont'd)

The updates of ALS can be written as

$$\mathbf{A}^{(i+1)} = \mathbf{Y}(\mathbf{B}^{(i)})^T(\mathbf{B}^{(i)}(\mathbf{B}^{(i)})^T)^{-1}$$
$$\mathbf{B}^{(i+1)} = ((\mathbf{A}^{(i+1)})^T\mathbf{A}^{(i+1)})^{-1}(\mathbf{A}^{(i+1)})^T\mathbf{Y}$$

- ALS is guaranteed to converge an optimal solution to $\min_{\mathbf{A},\mathbf{B}} \|\mathbf{Y} - \mathbf{AB}\|_F^2$ under some mild assumptions [2]

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

Low-Rank Matrix Completion

Aim: Given $\mathbf{Y} \in \mathbb{R}^{m \times n}$ 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, recover the missing entries of $\mathbf{Y}$

Applications: recommender system, data science, etc.

Example: Movie recommendation [3]

- $\mathbf{Y}$ records how user i likes movie j
- $\mathbf{Y}$ has lots of missing entries; A user doesn't watch all movies

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

- $\mathbf{Y}$ may be assumed to have low rank;
  Research shows that only a few factors affect users' preferences

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

ALS alternative for Low-Rank Matrix Completion

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

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

An ALS alternative for matrix completion:[4]

- Consider an equivalent reformulation of the problem

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

[4] 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.

ALS alternative for Low-Rank Matrix Completion (cont'd)

- Do alternating optimization according to the equivalent problem

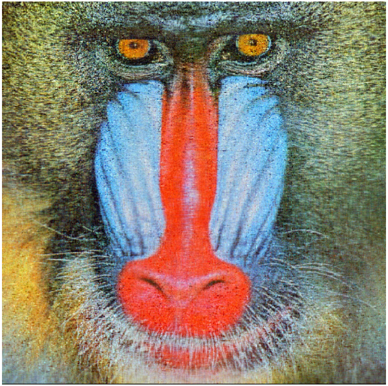
$$\begin{aligned} \mathbf{A}^{(i+1)} &= \arg \min_{\mathbf{A} \in \mathbb{R}^{m \times k}} \|\mathbf{Y} - \mathbf{A}\mathbf{B}^{(i)} - \mathbf{R}^{(i)}\|_F^2 \\ \mathbf{B}^{(i+1)} &= \arg \min_{\mathbf{B} \in \mathbb{R}^{k \times n}} \|\mathbf{Y} - \mathbf{A}^{(i+1)}\mathbf{B} - \mathbf{R}^{(i)}\|_F^2 \\ \mathbf{R}^{(i+1)} &= \arg \min_{\substack{\mathbf{R} \in \mathbb{R}^{m \times n} \\ r_{ij}=0,\ \forall (i,j) \in \Omega}} \|\mathbf{Y} - \mathbf{A}^{(i+1)}\mathbf{B}^{(i+1)} - \mathbf{R}\|_F^2 \end{aligned}$$

- The first two equations can be solved via LS as before

- The third equation has the closed-form solution

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

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. $k = 120$.

Beyond LS

- let $\tilde{\mathbf{a}}_i^T \in \mathbb{R}^{1\times n}$ denote the ith row of $\mathbf{A}$
  The LS problem can be rewritten as

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

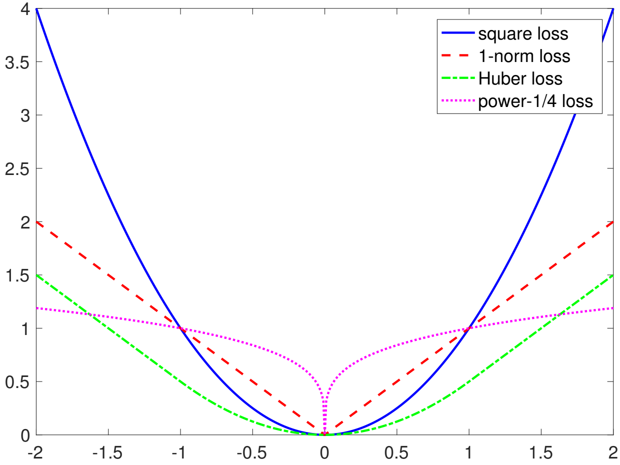
  where $\ell(z) = |z|^2$ is a loss function for measuring the badness of fit

- We can indeed use other loss functions such as
  - 1-norm loss: $\ell(z) = |z|$
  - Huber loss: $\ell(z) = \begin{cases} \frac{1}{2}|z|^2, & |z| \le 1 \\ |z| - \frac{1}{2}, & |z| > 1 \end{cases}$
  - power-p loss: $\ell(z) = |z|^p$, with $p < 1$

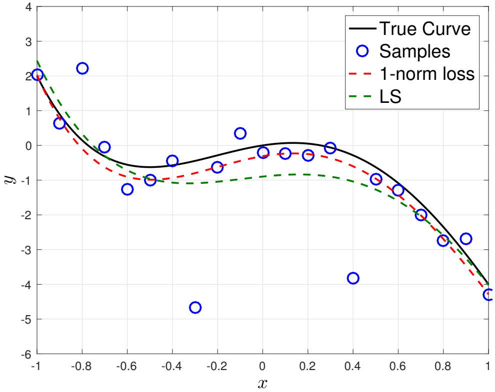
- The above loss functions are more robust against outliers

- However, they require optimization and don't result in a clean closed-form solution as LS

Illustration of Loss Functions



Example of Curve Fitting



"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.

Cheaper LS Solution

Recall that LS requires to solve the normal equation

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

Complexity: $\mathcal{O}(n^3)$

- We also need to compute $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{y}$, whose complexities are $\mathcal{O}(mn^2)$ and $\mathcal{O}(mn)$, respectively

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

We may acquire computationally less expensive LS solutions, with compromise of solution accuracy

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: 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

Convergence results:

- For convex f and with proper μ, gradient descent converges to an optimal solution
- For non-convex f and with proper μ, gradient descent converges to a stationary point

Gradient Descent (cont'd)

Gradient descent for LS:

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} - 2\mu(\mathbf{A}^T\mathbf{A}\mathbf{x}^{(k-1)} - \mathbf{A}^T\mathbf{y}), \quad k = 0, 1, \ldots$$

Complexity for dense $\mathbf{A}$:

- Computing $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{y}$: $\mathcal{O}(mn^2)$ and $\mathcal{O}(mn)$ (same as before)
  - $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{y}$ are cached for subsequent use
- Each iteration: $\mathcal{O}(n^2)$

Complexity for sparse $\mathbf{A}$:

- Computing $\mathbf{A}^T\mathbf{y}$: $\mathcal{O}(nnz(\mathbf{A}))$
- Each iteration: $\mathcal{O}(n + nnz(\mathbf{A}))$
  - $\mathbf{A}^T\mathbf{A}$ is not necessarily sparse, so we do $\mathbf{A}\mathbf{x}^{(k-1)}$ and then $\mathbf{A}^T(\mathbf{A}\mathbf{x}^{(k-1)})$

More advanced optimization methods can be applied (e.g., conjugate gradient method)

Online LS

Recall the LS formulation

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

Originally, the solving of LS is a batch process, i.e., solve one $\mathbf{x}$ given the whole $(\mathbf{A}, \mathbf{y})$

In many applications, each $(\tilde{\mathbf{a}}_t, y_t)$ comes as time t goes
We want the solving process to be adaptive/in real time

Incremental Gradient Descent for Online LS

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 stochastic gradient descent, least mean squares (LMS) (in 70's)

Incremental gradient descent for LS:

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

- At each time t, only need the last iterate $\mathbf{x}_{t-1}$ and the current data $(\tilde{\mathbf{a}}_t, y_t)$

Recursive LS

Recursive LS (RLS) formulation:

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

where $0 < \lambda \leq 1$ is prescribed, called the forgetting factor

- Weigh the importance of $|\tilde{\mathbf{a}}_i^T \mathbf{x} - y_i|^2$ w.r.t. time t: The present is most important while distant pasts are insignificant
- How much we remember the past depends on λ

At first look, the RLS solution is $\mathbf{x}_t = \mathbf{R}_t^{-1}\mathbf{q}_t$ (assume $\mathbf{R}_t$ nonsingular), where

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

$\mathbf{x}_t$ can be derived recursively by using the Woodbury matrix identity and exploiting the problem structures

Woodbury Matrix Identity

For $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ with proper sizes,

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

assuming that the inverses above exist

For the RLS problem, it is sufficient to consider 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

$$\mathbf{R}_t = \lambda \mathbf{R}_{t-1} + \tilde{\mathbf{a}}_t \tilde{\mathbf{a}}_t^T, \quad \mathbf{q}_t = \lambda \mathbf{q}_{t-1} + y_t \tilde{\mathbf{a}}_t$$

Using the Woodbury matrix identity,

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

Let $\mathbf{P}_t = \mathbf{R}_t^{-1}$ and $\mathbf{g}_t = \dfrac{1}{1 + \frac{1}{\lambda}\tilde{\mathbf{a}}_t^T \mathbf{R}_{t-1}^{-1}\tilde{\mathbf{a}}_t}(\tfrac{1}{\lambda}\mathbf{R}_{t-1}^{-1}\tilde{\mathbf{a}}_t)$. Then,

$$\mathbf{g}_t = \frac{1}{1 + \frac{1}{\lambda}\tilde{\mathbf{a}}_t^T \mathbf{P}_{t-1}\tilde{\mathbf{a}}_t}(\tfrac{1}{\lambda}\mathbf{P}_{t-1}\tilde{\mathbf{a}}_t), \quad \mathbf{P}_t = \tfrac{1}{\lambda}\mathbf{P}_{t-1} - \mathbf{g}_t(\tfrac{1}{\lambda}\mathbf{P}_{t-1}\tilde{\mathbf{a}}_t)^T$$

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

Recursive LS

RLS recursion:

$$\mathbf{g}_t = \frac{1}{1 + \frac{1}{\lambda}\tilde{\mathbf{a}}_t^T \mathbf{P}_{t-1}\tilde{\mathbf{a}}_t}\left(\frac{1}{\lambda}\mathbf{P}_{t-1}\tilde{\mathbf{a}}_t\right)$$

$$\mathbf{P}_t = \frac{1}{\lambda}\mathbf{P}_{t-1} - \mathbf{g}_t\left(\frac{1}{\lambda}\mathbf{P}_{t-1}\tilde{\mathbf{a}}_t\right)^T$$

$$\mathbf{x}_t = \mathbf{x}_{t-1} + (y_t - \tilde{\mathbf{a}}_t^T \mathbf{x}_{t-1})\mathbf{g}_t$$

Remarks:

- It replaces the term $2\mu\tilde{\mathbf{a}}_t$ in incremental gradient descent with $\mathbf{g}_t$
- The RLS recursion may be numerically unstable as empirical results suggested. Modified RLS schemes were developed to mend this issue