Matrix Methods in Signal Processing ...
(Lecture notes for EECS 551)

Jeff Fessler
University of Michigan

August 27, 2019
Chapter 0

EECS 551 Course introduction: F19

Contents (class version)

| 0.1 Course logistics                                      | 0.2 |
| 0.2 Julia language                                       | 0.9 |
| 0.3 Course topics                                        | 0.16|

These lecture notes initially were based extensively on Prof. Raj Nadakuditi’s hand-written notes. I am grateful to him for sharing his course materials. I also thank former GSIs David Hong and Steven Whitaker and 551 students in F17 and F18 for many corrections to earlier versions.

These notes were typeset using \LaTeX. One way to learn \LaTeX is to use http://overleaf.com.
0.1 Course logistics

EECS 551: Matrix Methods For Signal Processing, Data Analysis & Machine Learning

Lecture: Tue, Thu 9-10:30AM, 1500 EECS
Discussion: 012 Fri. 9:30-10:30 AM, 1003 EECS
011 Fri. 10:30-11:30 AM, 1500 EECS

Instructor: Prof. Jeff Fessler fessler@umich.edu https://web.eecs.umich.edu/~fessler/
Office hours: Wed, Thu 10:30-11:30AM (often until 11:45AM), 4431 EECS.
Include [eecs551-f19] in email subject for less slow response. Use Canvas/Piazza when possible.

GSI (office hours held in 3312 EECS):
• Caroline Crockett cecroc@umich.edu http://web.eecs.umich.edu/~cecrop
  Mon 4-5PM Tue 10:30-12PM Thu 2-3PM

Course materials:

• Primary site is Canvas: https://umich.instructure.com/courses/310523
  (homework, solutions, lecture notes, announcements, etc.)
• Annotated versions of class notes: https://tinyurl.com/f19-551-lecture
• Secondary site (demos, back-ups): http://web.eecs.umich.edu/~fessler/course/551

Course goal: provide a mathematical foundation for subsequent signal processing and machine learning courses, while also introducing matrix-based SP/ML methods that are useful in their own right.
Prerequisites

DSP (i.e., EECS 351, formerly 451) or graduate standing.
Prof. Nadakuditi’s EECS 505 perhaps relies less on DSP background.

Exams

Midterm Exam 1: Mon. Oct. 21, 6-8:00 PM, 1303&1500 EECS
Midterm Exam 2: Mon. Nov. 25, 6-8:00 PM, 1303&1500 EECS
Final Exam: Wed. Dec 18, 1:30-3:30 PM, Room TBA

Grades

Homework and task sheets 20%
Clicker 5%
Canvas quizzes 0% (must attempt by deadline to view later!)
Midterm exam 1 20%
Midterm exam 2 25%
Final exam 30%

Final grade cutoffs will be 90/80/70% or lower. Exam scores may be standardized. Grade history.

Honor code

The UM College of Engineering Honor Code applies. See collaboration policies below. See https://ossa.engin.umich.edu/honor-council/ for details.
Homework

Typically due on Thursday at 4PM. Typically an automatic extension to Friday at 4PM. No further.
Submit scans of solutions to [https://gradescope.com](https://gradescope.com). (HW1 on Canvas!)
Hopefully will be graded and “returned” via gradescope within a week.
Written regrade requests via gradescope within 3 days of return date.

Actions:

- Check for your name on gradescope (should be there thanks to Canvas integration)
- Review gradescope scan/pdf submission process. There are also video instructions.

Collaboration policy: Homework assignments are to be completed on your own. You are allowed to consult with other students (and instructors) during the conceptualization of a solution, but all written work, whether in scrap or final form, is to be generated by you, working alone. Also, you are not allowed to use, or in any way derive advantage from, the existence of solutions prepared in prior years. Violation of this policy is an honor code violation. If you have questions about this policy, please contact me. While collaboration can sometimes be helpful to learning, if overused, it can inhibit the development of your problem solving skills.

Ethics

Sharing any materials from this class with other individuals not in the class without written instructor permission will be treated as an Honor Code violation. Posting your own solutions (including code) on public sites like github.com is also prohibited. Keep your materials private! In particular, uploading any materials from this class to web sites akin to coursehero.com will be reported to the Honor Council.
Homework grading

Homework grading is constrained by GEO union policies. See:
http://web.eecs.umich.edu/~fessler/course/551/r/grading,geo.txt
http://web.eecs.umich.edu/~fessler/course/551/r/grader-duties.pdf

Manually graded problems will be on a scale of 0-3:
0. No solution was attempted
1. A solution was attempted but the approach used did not recognizably conform to any in the solution set
2. The approach used recognizably conformed to one in the solution set, but the answer was incorrect.
3. A solution approach recognizably conformed to one in the solution set, and the answer was correct.

JULIA-based auto-graded problems (details on HW1) typically will be 10 points each (10 or 0).

Quizzes

Starting in the second week of the course, there will be short quizzes (typically 4-6 questions) on Canvas that are due by 9am on Tue. and Thu. The quizzes will become available 24 hours before each class. These are designed to be quick checks of your understanding of the material covered up to that point. You have 10 minutes to complete the quiz. Their main purpose is learning, not assessment. Thus, Canvas shows you the answers right after you take it.

Post concerns about any quiz question to a Canvas Discussion after the Quiz due date. The instructor quiz interface on Canvas is horrid, so quizzes and lectures can get out of sync. That is why quizzes are worth 0%. Apparently Canvas only allows a student to view a past quiz (e.g., for exam review) that they attempted!
Missing class

- Classes are captured/recorded and viewable on Canvas.
- Missed clicker questions and quizzes cannot be made up.
- Annotated notes are available online; see p. 0.2.
- Daily topic list: http://web.eecs.umich.edu/~fessler/course/551 (topics)

Books and other resources

Action: Decide whether to buy reference textbook [1]:
Laub, 2005; Matrix analysis for scientists and engineers.
Should be on reserve at UM Engineering Library

$52 at http://bookstore.siam.org/ot91 - 30% member discount.
Student membership is free: https://siam.org/students/memberships.php
(Select “University of Michigan” not “UM Ann Arbor” as the Academic Member Institution.”)

Khan Academy: http://www.khanacademy.org/math/linear-algebra
Free online book about JULIA:
Clickers

http://caenfaq.engin.umich.edu/10909-clickers/

Bring batteries!

Action: Buy at http://computershowcase.umich.edu/remotes/ ($29 used, buy back for $19)

Action: Register your clicker at Canvas.

Clicker question scoring: 2 points for answering, 3 points for correct answer. (Learning, not assessment.)

PDF lecture note features

These notes highlight some important terms in red.
Many important terms have links in the pdf documents to Wikipedia in violet. Some links look like: [wiki]
Those links are clickable in the pdf and should cause your browser to open at the appropriate url.

Define. Key definitions are shaded like this.

Particularly important topics are shaded like this.

JULIA code is shaded like this.

Boxes with this color need completion during class.

A road hazard or “dangerous bend” symbol in the margin warns of tricky material.

A double diamond symbol is “experts only” material included for reference that is likely beyond the scope of EECS 551 exams.

These notes are not a textbook; they are designed for classroom use. My goal is that every other page or so (except in this part) should have something more interactive than just text, such as a figure or a clicker question or a JULIA code snippet or some incomplete equation(s) that students must complete during class.

These notes are formatted with 16:10 aspect ratio to match the projector in the lecture room; that format is well-suited for printing two slides per paper side. If you print, please save paper by using that option.

Action: Print the next chapter (not this one) or bring pdf to class on a suitable device for annotating.
0.2 Julia language

All code examples and homework code templates will use the Julia programming language.

Why?

- It is free; you can download from https://julialang.org
- It is a real programming language, developed for numerical computing [10].
- Prof. Nadakuditi and I have used it since W17 for multiple courses at UM and MIT (505, 551, 598...)
- Interactive (like Python and MATLAB), yet fast execution because it is compiled.
- Much of its syntax is like MATLAB, so much easier for me to learn and use than python.
  
  For differences with MATLAB, see: https://docs.julialang.org/en/v1/manual/noteworthy-differences
- DSP / data-science / machine learning are all done with many software languages...
- Jupyter notebooks (based on IPython) are educational, integrating math with documented code and figures.
- JuliaBox allows within-browser use in the cloud, mostly for limited toy experimenting. Not recommended!

Some documentation / books:

• Online “Getting started with JULIA” book: https://search.lib.umich.edu/catalog/record/013714926
• Cheatsheet: https://cheatsheets.quantecon.org/julia-cheatsheet.html
• Cheatsheet: http://math.mit.edu/~stevenj/Julia-cheatsheet.pdf
• YouTube intro video: https://www.youtube.com/watch?v=puMIBYthsjQ

News articles about business uses:
• Forbes magazine article
• InfoWorld comparison of JULIA and Python
• Nature article about JULIA

Sponsors of Juliacon 2018 and Juliacon 2019:
## A brief comparison of three interactive languages

<table>
<thead>
<tr>
<th>Operation</th>
<th>MATLAB</th>
<th>JULIA</th>
<th>Python</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dot product</td>
<td><code>dot(x,y)</code></td>
<td><code>dot(x,y)</code></td>
<td><code>np.dot(x,y)</code></td>
</tr>
<tr>
<td>Matrix mult.</td>
<td><code>A * B</code></td>
<td><code>A * B</code></td>
<td><code>A @ B</code></td>
</tr>
<tr>
<td>Element-wise</td>
<td><code>A .* B</code></td>
<td><code>A .* B</code></td>
<td><code>A * B</code></td>
</tr>
<tr>
<td>Scaling</td>
<td><code>3 * A</code></td>
<td><code>3A or 3*A</code></td>
<td><code>3 * A</code></td>
</tr>
<tr>
<td>Matrix power</td>
<td><code>A^2</code></td>
<td><code>A^2</code></td>
<td><code>np.linalg.matrix_power(A,2)</code></td>
</tr>
<tr>
<td>Element-wise</td>
<td><code>A.^2</code></td>
<td><code>A.^2</code></td>
<td><code>A**2</code></td>
</tr>
<tr>
<td>Inverse</td>
<td><code>inv(A)</code></td>
<td><code>inv(A)</code></td>
<td><code>np.linalg.inv(A)</code></td>
</tr>
<tr>
<td>Inverse</td>
<td><code>A^(-1)</code></td>
<td><code>A^(-1)</code></td>
<td><code>np.linalg.inv(A)</code></td>
</tr>
<tr>
<td>Range</td>
<td><code>1:9</code></td>
<td><code>1:9</code></td>
<td><code>np.arange(1,9,1)</code></td>
</tr>
<tr>
<td>Range</td>
<td><code>linspace(0,4,9)</code></td>
<td><code>LinRange(0,4,9)</code></td>
<td><code>np.arange(0,4.01,0.5)</code></td>
</tr>
<tr>
<td>Strings</td>
<td>'text'</td>
<td>&quot;text&quot;</td>
<td>(either)</td>
</tr>
<tr>
<td>Inline func.</td>
<td><code>f = @(x,y) x+y</code></td>
<td><code>f = (x,y) -&gt; x+y</code></td>
<td><code>f = lambda x,y : x+y</code></td>
</tr>
<tr>
<td>Increment</td>
<td><code>A = A + B</code></td>
<td><code>A += B</code></td>
<td><code>A += B</code></td>
</tr>
<tr>
<td>Herm. transp.</td>
<td><code>A'</code></td>
<td><code>A'.T</code></td>
<td><code>A.conj().T</code></td>
</tr>
<tr>
<td></td>
<td><code>\begin{bmatrix} 1 &amp; 2 \\ 3 &amp; 4 \end{bmatrix}</code></td>
<td><code>\begin{bmatrix} 1 &amp; 2; 3 &amp; 4 \end{bmatrix}</code></td>
<td><code>np.array([[1, 2], [3, 4]])</code></td>
</tr>
</tbody>
</table>

The JULIA column assumes you have typed: `using LinearAlgebra` for using the `dot` function. See [https://cheatsheets.quantecon.org/](https://cheatsheets.quantecon.org/) for more.
**JULIA logistics**

- Software parts of homework solutions (JULIA code) will be auto-graded, with *unlimited* tries.
- More details in Discussion section for HW
- Some tutorials (and cautionary notes for MATLAB users):
  - [https://web.eecs.umich.edu/~fessler/course/551/julia/tutor/](https://web.eecs.umich.edu/~fessler/course/551/julia/tutor/)
    - julia-tutor-vector: vector/matrix operations and *call by reference* aspect of JULIA
    - julia-tutor-slice: matrix indexing (slicing)
    - julia-tutor-sum-simd.html: acceleration using @simd
- Later we will do in-class activities using JULIA and you will want to bring a laptop with JULIA on it.
- Editors (see list at JULIA web site: [https://julialang.org/](https://julialang.org/))
  - Atom: [https://atom.io/](https://atom.io/)
  - vim: [https://www.vim.org/](https://www.vim.org/)
- **Use JULIA version 1.1 or later for F19; current version is 1.2.** Beware of online Q/A for older versions of JULIA!
- **Actions:** Bring laptop to Discussion section Friday.
One professor’s software language history

<table>
<thead>
<tr>
<th>year</th>
<th>name</th>
<th>still using?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1977</td>
<td>BASIC</td>
<td></td>
</tr>
<tr>
<td>1980</td>
<td>FORTRAN</td>
<td></td>
</tr>
<tr>
<td>1981</td>
<td>Z80 assembly</td>
<td></td>
</tr>
<tr>
<td>1982</td>
<td>APL</td>
<td></td>
</tr>
<tr>
<td>1983</td>
<td>PASCAL</td>
<td></td>
</tr>
<tr>
<td>1983</td>
<td>C</td>
<td>Y?</td>
</tr>
<tr>
<td>1985</td>
<td>LISP</td>
<td></td>
</tr>
<tr>
<td>1986</td>
<td>CSH</td>
<td>Y</td>
</tr>
<tr>
<td>1988</td>
<td>Matlab</td>
<td>Y?</td>
</tr>
<tr>
<td>2000</td>
<td>Perl</td>
<td></td>
</tr>
<tr>
<td>2004</td>
<td>Python</td>
<td></td>
</tr>
<tr>
<td>2010</td>
<td>CUDA</td>
<td>Y?</td>
</tr>
<tr>
<td>2017</td>
<td>Julia</td>
<td>Y</td>
</tr>
</tbody>
</table>

**JULIA** has a **machine-learning** library called **Flux** [11].
It also has an interface to **Tensorflow** if you prefer that.
Another “ML in JULIA” package is **MLJ**.
There is also a CUDA interface for GPU programming [12].

**MATLAB** is “free” for UM students:

http://caenfaq.engin.umich.edu/10378-Free-Software-for-Students/matlab-for-students
**JULIA: getting started**

For F19, we recommend (but do not require) that you use the Juno IDE in the powerful Atom editor because of its excellent integration with the Julia Debugger. Alternatively, you may use your own favorite editor (though you may find debugging more challenging) or use the (free) JuliaPRO version from https://juliacomputing.com/products/juliapro

- **Actions:** Follow detailed installation instructions at  
  https://github.com/JeffFessler/MIRT.jl/blob/master/doc/start-juno.md
- At the JULIA prompt, try launching a Jupyter notebook:
  ```julia
  using IJulia; notebook()
  ```
  (Could be slow the first time as it gets compiled.)
  For help with Jupyter, see https://github.com/JuliaLang/IJulia.jl
  e.g., you might prefer `notebook(detached=true)`
  or `notebook(detached=true, dir="/some/path")`
- Experiment with the Jupyter notebook, and peruse some online resources.
- For documentation of the Plots.jl plotting package, see:
  http://docs.juliaplots.org/latest/
- Link to video about Juno debugger (20 mins into JuliaCon 2019 talk)
  https://youtu.be/SU0SmQnnGys?t=1200
Clicker survey questions

How are you feeling about JULIA? (Pick the answer that is your strongest feeling.)
A: Anticipate it will be useful
B: Bothered about learning something new
C: Concerned about my software skills
D: Indifferent (or none of the others apply)
E: Excited to be on the cutting edge of numerical computing

Prior software experience?
A: Not Matlab, but any of C or C++ or Python
B: Matlab only
C: Matlab and (Python | C | C++)
D: JULIA and (Matlab | Python | C | C++)
E: None of the above

Office hours (do not answer if your schedule is still uncertain)
A: Wed 10:30-11:30 works for me, but not Thu.
B: Thu 10:30-11:30 works for me, but not Wed.
C: Both Wed and Thu work
D: Neither Wed nor Thu work for me, but some GSI hour(s) work
E: None of the Prof. or GSI office hours work for me
# 0.3 Course topics

Here are some likely course topics in approximate chronological order, along with sections from [1].

1. Introduction to matrices

<table>
<thead>
<tr>
<th>Topic (Laub §)</th>
<th>Finite difference</th>
<th>Hermitian</th>
<th>invertible matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Julia</td>
<td>pentadiagonal</td>
<td>linear algebra</td>
<td>Laplace’s formula</td>
</tr>
<tr>
<td>vector (1.1)</td>
<td>Gram matrix</td>
<td>dot product</td>
<td>eigenvalues (9.1)</td>
</tr>
<tr>
<td>matrix (1.1)</td>
<td>upper Hessenberg</td>
<td>inner product</td>
<td>characteristic polynomial</td>
</tr>
<tr>
<td>linear operation</td>
<td>eigenvalue algorithms</td>
<td>outer product</td>
<td>fundamental theorem of algebra</td>
</tr>
<tr>
<td>DFT</td>
<td>lower Hessenberg</td>
<td>rank 1 matrix</td>
<td>Gram matrix</td>
</tr>
<tr>
<td>system of linear equations</td>
<td>rectangular diagonal</td>
<td>matrix multiplication</td>
<td>covariance matrix</td>
</tr>
<tr>
<td>convolution</td>
<td>dense matrix</td>
<td>(1.2)</td>
<td>singular matrix</td>
</tr>
<tr>
<td>LTI</td>
<td>sparse matrix</td>
<td>orthogonal (1.3)</td>
<td>scatter matrix</td>
</tr>
<tr>
<td>term-document matrix</td>
<td>Toeplitz</td>
<td>orthonormal</td>
<td>trace (1.1)</td>
</tr>
<tr>
<td>diagonal</td>
<td>Circulant</td>
<td>norm</td>
<td>field (2.1)</td>
</tr>
<tr>
<td>upper triangular</td>
<td>block matrix</td>
<td>orthogonal matrix</td>
<td>vector space (2.1)</td>
</tr>
<tr>
<td>Gaussian elimination</td>
<td>block diagonal matrix</td>
<td>unitary matrix</td>
<td>linear transform (3.1)</td>
</tr>
<tr>
<td>lower triangular</td>
<td>transpose</td>
<td>Parseval’s theorem</td>
<td></td>
</tr>
<tr>
<td>Cholesky decomposition</td>
<td>Hermitian transpose</td>
<td>determinant (1.4)</td>
<td></td>
</tr>
<tr>
<td>tridiagonal</td>
<td>symmetric</td>
<td>Gram matrix</td>
<td></td>
</tr>
</tbody>
</table>
# 2. Matrix factorizations / SVD

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>eigenvalues</td>
<td>(10.1)</td>
</tr>
<tr>
<td>spectral theorem</td>
<td></td>
</tr>
<tr>
<td>eigenvectors</td>
<td></td>
</tr>
<tr>
<td>orthogonal</td>
<td></td>
</tr>
<tr>
<td>orthonormal</td>
<td></td>
</tr>
<tr>
<td>eigendecomposition</td>
<td></td>
</tr>
<tr>
<td>diagonalizable</td>
<td></td>
</tr>
<tr>
<td>orthogonal</td>
<td></td>
</tr>
<tr>
<td>spectral norm</td>
<td>(7.4)</td>
</tr>
<tr>
<td>MIMO channels</td>
<td></td>
</tr>
<tr>
<td>beamforming</td>
<td></td>
</tr>
<tr>
<td>positive definite</td>
<td>(10.2)</td>
</tr>
</tbody>
</table>

# 3. Subspaces and rank

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>dimensionality reduction</td>
<td></td>
</tr>
<tr>
<td>subspace</td>
<td>(2.2)</td>
</tr>
<tr>
<td>periodic functions</td>
<td></td>
</tr>
<tr>
<td>span</td>
<td>(2.3)</td>
</tr>
<tr>
<td>linear combinations</td>
<td></td>
</tr>
<tr>
<td>linearly independent</td>
<td>(2.3)</td>
</tr>
<tr>
<td>linearly dependent</td>
<td></td>
</tr>
<tr>
<td>monomials</td>
<td></td>
</tr>
<tr>
<td>basis</td>
<td>(2.3)</td>
</tr>
<tr>
<td>discrete cosine transform</td>
<td></td>
</tr>
<tr>
<td>wavelet transform</td>
<td></td>
</tr>
<tr>
<td>coordinate system</td>
<td></td>
</tr>
<tr>
<td>additive synthesis</td>
<td></td>
</tr>
<tr>
<td>basis</td>
<td>(2.3)</td>
</tr>
<tr>
<td>subspace sum</td>
<td>(2.4)</td>
</tr>
<tr>
<td>intersection</td>
<td></td>
</tr>
<tr>
<td>direct sum</td>
<td></td>
</tr>
<tr>
<td>orth. complement</td>
<td>(3.4)</td>
</tr>
<tr>
<td>linear map</td>
<td>(3.1)</td>
</tr>
<tr>
<td>range</td>
<td>(3.4)</td>
</tr>
<tr>
<td>column space</td>
<td></td>
</tr>
<tr>
<td>row space</td>
<td></td>
</tr>
<tr>
<td>rank</td>
<td>(3.5)</td>
</tr>
<tr>
<td>null space</td>
<td>(3.4)</td>
</tr>
<tr>
<td>kernel</td>
<td></td>
</tr>
<tr>
<td>nullity</td>
<td></td>
</tr>
<tr>
<td>fundamental theorem of linear algebra</td>
<td>(3.5)</td>
</tr>
<tr>
<td>orthogonal basis</td>
<td></td>
</tr>
</tbody>
</table>
4. Linear equations and least-squares

- linear equations (6.1)
- linear least-squares (8.1)
- residual
- orthogonal polynomials
- convex function
- normal equations (8.1)
- SVD (8.4)
- over-determined system
- Moore-Penrose pseudoinverse (4.1)
- left inverse (4.3)
- right inverse
- rectangular diagonal
- SVD (5.2)
- QR decomposition (8.5)
- under-determined
- orthogonality principle
- error
- linear variety
- flat
- idempotent matrix
- minimum norm sol. (8.1)
- compressed sensing
- truncated SVD
- floating-point precision
- condition number
- low-rank approx. (8.1)
- Tikhonov regularization
- regularization parameter
- conjugate gradient

5. Norms

- vector norm (7.3)
- Euclidean norm
- triangle inequality
- Parseval’s theorem
- inner product spaces (7.2)
- inner product
- Frobenius inner product
- parallelogram law
- Cauchy-Schwarz ineq.
- angle between subspaces
- correlation coefficient
- matrix norms (7.4)
- sub-multiplicative
- Frobenius norm
- induced norm
- Schatten p-norm
- norm equivalence
- unitarily invariant norms
- Procrustes problem
- polar decomposition
- idempotent matrix
### 6. Low-rank approximation

<table>
<thead>
<tr>
<th>Dimensionality reduction</th>
<th>Unitary invariance</th>
<th>estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low-rank approximation</td>
<td>PCA</td>
<td>weakly differentiable</td>
</tr>
<tr>
<td>Factor analysis</td>
<td>Multidimensional scaling</td>
<td>Divergence</td>
</tr>
<tr>
<td>Scree plot</td>
<td>Geodesic distances</td>
<td>OptShrink</td>
</tr>
<tr>
<td>Eckart-Young-Mirsky</td>
<td>Heaviside step function</td>
<td>matrix completion</td>
</tr>
<tr>
<td>Theorem</td>
<td>Stein’s unbiased risk</td>
<td>Netflix problem</td>
</tr>
</tbody>
</table>

### 7. Optimization basics

<table>
<thead>
<tr>
<th>Optimization</th>
<th>Matrix powers</th>
<th>Lipschitz continuity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convergence</td>
<td>Positive semidefinite</td>
<td>Taylor’s theorem</td>
</tr>
<tr>
<td>(Matrix) square root</td>
<td>Positive definite</td>
<td>Logistic regression</td>
</tr>
<tr>
<td>Principal square root</td>
<td>Gradient descent</td>
<td>Hessian matrix</td>
</tr>
</tbody>
</table>
8. Special matrices

monic polynomial  companion matrix  minimum polynomial
Vandermonde matrix  Kronecker sum  circulant matrix
DFT  fast Fourier transform

power iteration  positive matrix  nonnegative matrix
primitive matrix  Geršgorin disk theorem  Perron-Frobenius
theorem  Gelfand’s formula

multiplicity one  algebraic multiplicity  geometric multiplicity
irreducible matrices  Markov chain  directed graph
transition matrix  stochastic eigenvector

law of total probability  irreducible matrix  simple eigenvalue
strongly connected graph  Google’s PageRank

9. Matrix completion

matrix completion  ill posed  degrees of freedom
Netflix problem  sampling bounds
latent variable  polylog
Hadamard product

NP hard  projection onto convex set
majorize-minimize (MM)  ISTA
proximal gradient method  proximity operation
Many applications in signal processing and machine learning, especially in HW and discussion section.

Example. **photometric stereo** (shape from shading):

Example. Hand-written digit recognition
Bibliography


