Laporan Tugas Besar 2

IF2123 Aljabar Linier dan Geometri

Aplikasi Nilai Eigen dan Vektor Eigen dalam Kompresi Gambar

Kelompok 48 :

1. **Ken Kalang Al Qalyubi (13520010)**
2. **Farrel Ahmad (13520110)**
3. **Muhammad Gerald Akbar Giffera (13520143)**

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

Bab 1
Deskripsi Masalah

Gambar merupakan cara yang sangat penting untuk menyebarkan informasi di zaman digital ini, akan tetapi seringkali gambar-gambar tersebut memakan memori/tempat yang terlalu banyak sehingga akan muncul masalah jika ingin disebarkan atau dikirim. Oleh karena itu dilakukanlah kompresi gambar yang bertujuan untuk mengurangi ukuran gambar di memori tanpa mengubah bentuk gambar tersebut.

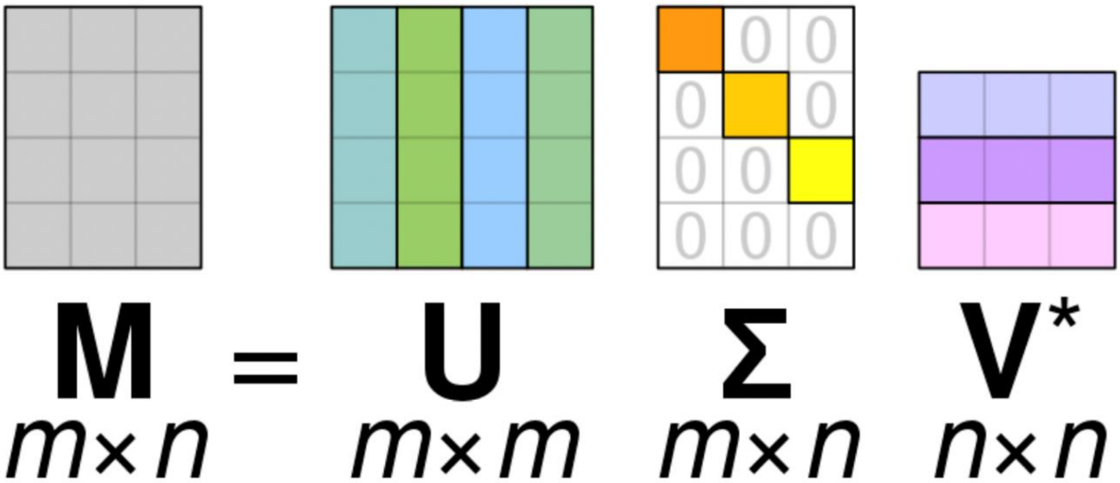
Di dalam Tugas Besar 2 ini, buatlah website yang bisa melakukan kompresi gambar dan menerima input file gambar lalu menampilkan gambar hasil kompresi, gunakan metode Singular Value Decomposition(SVD) dalam proses kompresi.

Bab 2

Teori Singkat

1. Singular Value Decomposition (SVD)

Singular Value Decomposition(SVD) adalah metode yang bisa digunakan untuk memfaktorkan sebuah matriks menjadi 3 buah matriks dengan memanfaatkan nilai eigen, vektor eigen, dan nilai singular dari sebuah matriks. Lebih tepatnya SVD digunakan untuk memfaktorkan matriks non persegi atau matriks berukuran m x n. Sebuah matriks akan terdekomposisi menjadi 3 buah matriks seperti dibawah berikut:



M: Matriks awal berukuran m x n

Untuk mendapatkan asing-masing matriks hasil dekomposisi yaitu U, ∑, dan V akan digunakan metode yang berbeda, akan tetapi sebelum masuk kedalam metode tersebut ada beberapa hal yang harus dipahami yaitu matriks orthogonal,nilai eigen, vector eigen, dan nilai singular.

a. Matriks ortogonal

Matriks ortogonal adalah matriks yang kolomnya terdiri dari vektor-vektor yang ortogonal satu sama lain (hasil *dot product* = 0), jika vektor-vektor kolom dari matriks tersebut merupakan vektor satuan (||v|| = 1) maka matriks tersebut disebut juga matriks **ortonormal**. Selain itu sebuah matriks A juga dapat dikatakan ortogonal jika memenuhi A^TA = I, dimana I adalah matriks

Identitas.

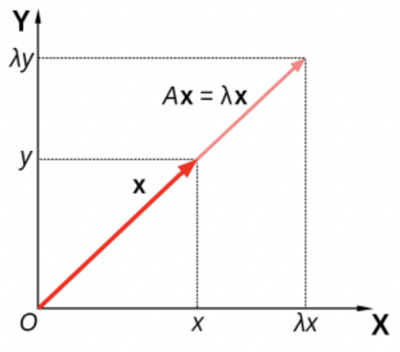
$$Q^TQ = \begin{pmatrix} 1/3 & 2/3 & -2/3 \\ -2/3 & 2/3 & 1/3 \\ 2/3 & 1/3 & 2/3 \end{pmatrix} \begin{pmatrix} 1/3 & -2/3 & 2/3 \\ 2/3 & 2/3 & 1/3 \\ -2/3 & 1/3 & 2/3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$Q^TQ = \begin{bmatrix} 3/7 & 2/7 & 6/7 \\ -6/7 & 3/7 & 2/7 \\ 2/7 & 6/7 & -3/7 \end{bmatrix} \begin{bmatrix} 3/7 & 2/7 & 6/7 \\ -6/7 & 3/7 & 2/7 \\ 2/7 & 6/7 & -3/7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Dari contoh diatas maka bisa disimpulkan bahwa matriks Q merupakan matriks ortogonal.

b. Nilai dan Vektor eigen

Sebuah vektor x disebut sebagai **vektor eigen** dari sebuah matriks jika hasil dari perkalian $Ax = \lambda x$ dimana λ adalah **nilai eigen**. Vector eigen bisa juga diartikan sebagai vector kolom yang jika dikalikan matriks n x n akan menghasilkan kelipatan dari vector itu sendiri.



Nilai eigen bisa dicari dengan persamaan berikut:

$$\begin{aligned} A\mathbf{x} &= \lambda\mathbf{x} \\ IA\mathbf{x} &= \lambda I\mathbf{x} \\ A\mathbf{x} &= \lambda I\mathbf{x} \\ (\lambda I - A)\mathbf{x} &= 0 \end{aligned}$$

Sehingga dari persamaan diatas bisa disimpulkan bahwa det (λI – A)x = 0, dimana I adalah matriks identitas. Sebagai contoh perhitungan nilai eigen bisa dilihat dibawah ini:

Tentukan nilai eigen dan vektor eigen dari matriks $A = \begin{bmatrix} 3 & 0 \\ 8 & -1 \end{bmatrix}$

$$\lambda I - A = \lambda\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 3 & 0 \\ 8 & -1 \end{bmatrix} = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} - \begin{bmatrix} 3 & 0 \\ 8 & -1 \end{bmatrix} = \begin{bmatrix} \lambda - 3 & 0 \\ -8 & \lambda + 1 \end{bmatrix}$$

$$\det(\lambda I - A) = 0 \rightarrow \begin{vmatrix} \lambda - 3 & 0 \\ -8 & \lambda + 1 \end{vmatrix} = 0 \rightarrow (\lambda - 3)(\lambda + 1) = 0 \rightarrow \text{persamaan karakteristik}$$

$$\rightarrow \lambda_1 = 3 \text{ dan } \lambda_2 = -1$$

Jadi, nilai-nilai eigen dari matriks A adalah $\lambda = 3$ dan $\lambda = -1$.

Sedangkan vektor eigen bisa ditemukan dengan cara mensubstitusikan λ dengan setiap nilai eigen yang didapatkan lalu menyelesaikan solusinya menggunakan metode eliminasi gauss. Bisa dilihat contoh perhitungan vector eigen untuk soal yang sama dibawah ini:

$$(\lambda I - A)\mathbf{x} = 0 \rightarrow \begin{bmatrix} \lambda - 3 & 0 \\ -8 & \lambda + 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Untuk $\lambda = 3 \rightarrow \begin{bmatrix} 0 & 0 \\ -8 & 4 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \rightarrow -8x_1 + 4x_2 = 0 \rightarrow 8x_1 = 4x_2 \rightarrow x_1 = ½\, x_2$

$\rightarrow$ Solusi: $x_1 = ½\, t$, $x_2 = t$, $t \in \mathbf{R}$

Vektor eigen: $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}t \\ t \end{bmatrix} = t\begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}$ $\rightarrow$membentuk **ruang eigen** (*eigenspace*)

Jadi, $\begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}$ adalah basis untuk ruang eigen dengan $\lambda = 3$

Ruang eigen ditulis sebagai $E(3) = \{ \mathbf{x} = t\begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}, t \in \mathbf{R} \}$

Bisa dilihat di atas merupakan contoh perhitungan dengan nilai λ = 3, hal ini harus diulang untuk nilai λ yang lainnya. Jika λ = -1 maka akan didapatkan

Ruang eigen ditulis sebagai $E(-1) = \{ \mathbf{x} = t\begin{bmatrix} 0 \\ 1 \end{bmatrix}, t \in \mathbf{R} \}$

c. Nilai singular

Misalkan A adalah matriks m x n. Jika $\lambda 1$, $\lambda 2$, …, λn adalah nilai-nilai eigen dari A TA, maka $\sigma 1 = \sqrt{\lambda 1}$, $\sigma 2 = \sqrt{\lambda 2}$, …, $\sigma n = \sqrt{\lambda n}$ disebut nilai singular dari matriks A,

Teorema: jika A adalah matriks m x n maka

i. ATA dapat terdiagonalisasi secara orthogonal
ii. Nilai eigen dari ATA non-negatif.

Lalu Langkah selanjutnya adalah mencari factor-faktor dari matriks mxn yaitu U, $\sum$, dan V, secara garis besar dapat dilakukan dengan 5 langkah berikut:

a. Untuk vektor singular kiri, hitung nilai-nilai eigen dari AAT . Rank(A) = k = banyaknya nilai-nilai eigen tidak nol dari AAT .
b. Tentukan vektor-vektor eigen u1 , u2 , …, um yang berkoresponden dengan nilainilai eigen dari AAT . Normalisasi u1 , u2 , …, um dengan cara setiap komponen vektornya dibagi dengan panjang vektor. Diperoleh matriks U.
c. 3. Untuk vektor singular kanan, hitung nilai-nilai eigen dari A TA lalu tentukan nilainilai-singularnya.
d. 4. Tentukan vektor-vektor eigen v1 , v2 , …, vn yang berkoresponden dengan nilainilai eigen dari A TA. Normalisasi v1 , v2 , …, vn dengan cara setiap komponen vektornya dibagi dengan panjang vektor. Diperoleh matriks V. Transpose-kan matriks V sehingga menjadi V T .
e. 5. Bentuklah matriks $\sum$ berukuran m x n dengan elemen-elemen diagonalnya adalah nilai-nilai singular dari matriks A dengan susunan dari besar ke kecil. Nilai singular di dalam $\sum$ adalah akar pangkat dua dari nilai-nilai eigen yang tidak nol dari A TA.

2. QR Decomposition

QR Decomposition merupakan salah satu metode pemfaktoran matriks selain SVD, pada QR Decomposition sebuah matriks A akan difaktorkan menjadi dua buah matriks yaitu Q dan R dengan persamaan berikut:

A = QR

Dimana Q adalah matriks orthogonal dan R merupakan matriks segitiga atas. Ada beberapa metode bisa digunakan untuk melakukan QR decomposition, salah satunya yang akan kami bahas adalah Gram-Schimdt process.

Pada proses ini matriks A direpresentasikan sebagai kumpulan dari vector kolom seperti berikut ini:

$$A = \left[\, \mathbf{a}_1 \mid \mathbf{a}_2 \mid \cdots \mid \mathbf{a}_n \,\right]$$

Dimana a1,a2,…an adalah vector kolom lalu proses selanjutnya bisa dirumuskan sebagai berikut:

$$\begin{aligned}
\mathbf{u}_1 &= \mathbf{a}_1, \qquad \mathbf{e}_1 = \frac{\mathbf{u}_1}{||\mathbf{u}_1||}, \\
\mathbf{u}_2 &= \mathbf{a}_2 - (\mathbf{a}_2 \cdot \mathbf{e}_1)\mathbf{e}_1, \qquad \mathbf{e}_2 = \frac{\mathbf{u}_2}{||\mathbf{u}_2||}. \\
\mathbf{u}_{k+1} &= \mathbf{a}_{k+1} - (\mathbf{a}_{k+1} \cdot \mathbf{e}_1)\mathbf{e}_1 - \cdots - (\mathbf{a}_{k+1} \cdot \mathbf{e}_k)\mathbf{e}_k, \qquad \mathbf{e}_{k+1} = \frac{\mathbf{u}_{k+1}}{||\mathbf{u}_{k+1}||}.
\end{aligned}$$

Hasil dari e1,e2,…en adalah vector kolom dari Q dan untuk setiap baris i dan kolom j di matriks R dapat dihitung dengan cara ai x ej, agar lebih mudah dimengerti bisa dilihat pada contoh dibawah ini

$$A = \left[\, \mathbf{a}_1 \mid \mathbf{a}_2 \mid \cdots \mid \mathbf{a}_n \,\right] = \left[\, \mathbf{e}_1 \mid \mathbf{e}_2 \mid \cdots \mid \mathbf{e}_n \,\right] \begin{bmatrix} \mathbf{a}_1 \cdot \mathbf{e}_1 & \mathbf{a}_2 \cdot \mathbf{e}_1 & \cdots & \mathbf{a}_n \cdot \mathbf{e}_1 \\ 0 & \mathbf{a}_2 \cdot \mathbf{e}_2 & \cdots & \mathbf{a}_n \cdot \mathbf{e}_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{a}_n \cdot \mathbf{e}_n \end{bmatrix} = QR.$$

Bab 3
Implementasi Pustaka dan Program dalam Python

1. Garis Besar Program

   Program kompresi gambar ini diimplementasikan kedalam bentuk situs web, sehingga untuk membuatnya kita membagi program menjadi dua bagian yaitu backend dan frontend. Backend berisi proses penerimaan data berbentuk gambar dan pemrosesan kompresi gambar, sedangkan frontend berisi tampilan situs web yang bisa dilihat oleh pengguna. Algoritma perhitungan eigen dan pengolahan matriks gambar sebagian menggunakan metode power iteration yang lengkapnya dapat dibaca di Daftar Referensi.

2. Library yang digunakan

   Situs Web ini dirancang dengan menggunakan Bahasa pemrograman python untuk bagian backend, sedangkan untuk frontend memanfaatkan HTML dan CSS. Berikut adalah library python yang kami gunakan untuk memudahkan pengerjaan program:

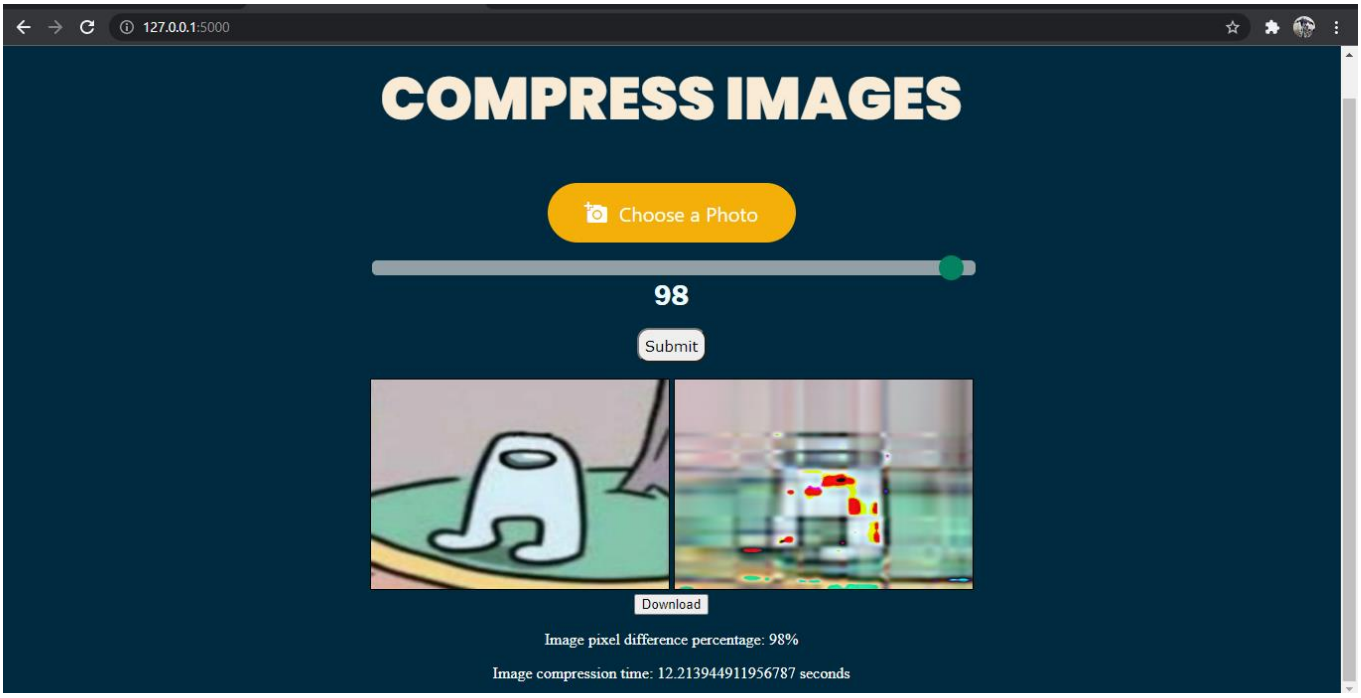
   a. Numpy: mempercepat operasi matriks dan kalkulasi, dan proses QR decomposition
   b. Sympy: untuk penggunaan komputasi eselon tereduksi
   c. Random: untuk menghasilkan nilai random dari 0 hingga 1
   d. Copy: untuk copy matriks
   e. Math: untuk menggunakan fitur akar(square root)
   f. cv2: library opencv-python untuk membaca sebuah gambar menjadi matriks BGR. Pada opencv, matriks RGB diformatkan dalam bentuk BGR.
   g. Flask: untuk backend website, menerima dan menampilkan gambar yang diterima dari user.

3. Fungsi/Prosedur yang di implementasikan

a. transpose(m), fungsi return matriks yang sudah ditranspose.
b. findEigen(m), fungsi return nilai-nilai eigen dari matriks.
c. findEigenVector(eig, m), fungsi return vektor-vektor eigen.
d. displayMatrix(m), prosedur menampilkan matriks (untuk debugging).
e. displaySol(arr), proseduremenampilkan solusi persamaan nilai eigen (untuk debugging).
f. rrEchelonForm(m), fungsi return matriks eselon tereduksi.
g. isRowZero(m,i), fungsi return boolean apakah baris i pada m adalah baris nol.
h. findBase(m), fungsi return nilai basis dari m.
i. findMatrixUandVt(A, k=None, epsilon=1e-10), fungsi return menghasilkan matrix U dan Vt SVD dengan mengambil k nya sebesar k.
j. findMatrixSigma(m), fungsi return matriks S (sigma) SVD.
k. normalize_vector(v), fungsi return normalisasi vektor.
l. ImgToMat(imgpath), fungsi return matriks dari file gambar.
m. filenamecvt(name), fungsi menghasilkan nama file untuk gambar yang telah dikompresi
n. randUnitVector(n), fungsi menghasilkan unit vector random yang bernilai float dari 0 sampai 1.
o. svd1d(A, epsilon=1e-10), fungsi perhitungan svd untuk matrix 1 dimensi dengan error maksimal epsilon.
p. compressImg(imgpath, percent_compression), prosedur menghasilkan gambar imgpath yang telah dikompresi sebesar percent_compression dan hasilnya di save.

Bab 4

Eksperimen



Gambar di atas adalah contoh penggunaan program pada website. Gambar kiri adalah gambar asli dan yang kiri adalah gambar hasil kompresi 98%. Kompresi 98% ini mengambil jumlah k sebanyak 4 atau k=4. Hasil kompresi menunjukkan bahwa gambar telah terkompresi membuat detail gambar menjadi kabur karena hanya mengambil rank matriks awal. Persentase kompresi yang besar membuat program semakin ringan karena jumlah k yang diambil lebih sedikit. Sedangkan kompresi 0% paling lama karena mengambil semua k dan algoritma tetap menghitung SVD-nya juga meskipun kompresinya 0% atau tidak terkompresi sama sekali. Semakin banyak k yang diambil atau semakin kecil persentase kompresi maka semakin berat komputasinya karena perhitugnan matriks juga lebih banyak. Selain itu, semakin besar resolusi gambar juga membuat komputasi semakin lama karena matriksnya juga semakin besar. Pada hasil percobaan tersebut juga ditampilkan bahwa membutuhkan waktu 12,21 detik untuk kompresi 98%.

Bab 5
Kesimpulan, Saran, dan Refleksi

1. Kesimpulan

   Berdasarkan hasil kinerja program yang telah dibuat dan deskripsi masalah, dapat diambil kesimbulan bahwa,

   - Program kompresi gambar menggunakan SVD yang telah dibuat cukup *power hungry* dan kecepatan komputasi masih cukup lambat
   - Kompresi gambar yang pixel nya besar akan memakan waktu komputasi jauh lebih lama karena matriks nya lebih besar.
   - Kompresi gambar dengan persentase kompresi kecil akan memakan waktu komputasi lebih lama.
   - Kompresi gambar yang dihasilkan membuat gambar terlihat terkompresi (menjadi kabur dan tidak detail), akan tetapi belum tentu membuat file gambar menjadi lebih kecil
2. Saran

   Saran yang bisa dilakukan untuk hasil program tersebut adalah,

   - Butuhnya pengetahuan kompleksitas algoritma lebih jauh untuk agar dapat membuat komputasi menjadi jauh lebih cepat
   - Membutuhkan pemahaman image processing lebih jauh untuk mengoptimalkan SVD atau membuat kompresi gambar dengan metode selain SVD yang mungkin jauh lebih efisien.
3. Refleksi

   Refleksi dari proses program yang telah dibuat adalah

   - Melalui program ini dapat dipelajari bahwa operasi matriks cukup memakan memori dan jumlah komputasi yang dihasilkan juga besar. Contohnya ketika dijalankan, laptop yang dipakai mengalami kinerja lebih berat untuk melakukan komputasi kompresi gambar
   - Melaui tugas besar ini dapat dipelajari bahwa informasi gambar banyak terletak pada nilai k atau rank awal-awal matriks-matriks SVD.

Daftar Referensi

Munir, Rinaldi. 2021. *Singular Value Decomposition*. Diakses pada 11 November 2021l https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-19b-Singular-value-decomposition.pdf

Yanovsky, Igor. *QR Decomposition with Gram-Schmidt*. Diakses pada 11 November 2021 https://www.math.ucla.edu/~yanovsky/Teaching/Math151B/handouts/GramSchmidt.pdf

Ciobanu, Andrei. 2021. *Computing Eigenvalues* and *Eigenvectors using QR Decomposition*. Diakses pada 12 November 2021 https://www.andreinc.net/2021/01/25/computing-eigenvalues-and-eigenvectors-using-qr-decomposition

ML Wiki. *Power Iteration*. Diakses pada 13 November 2021. http://mlwiki.org/index.php/Power_Iteration