Eigenvectors & eigenvalues

- Eigenvector e of a n x n matrix M
  - $e*M = \lambda*e$
  - Special vector - does not change direction
- Symmetric matrix $M=M^T$
- Orthogonal matrix $M*M^T=M^{T*}M=I$ ($M^{-1}=M^T$)
- M is real symmetric
  - e_i & λ_i are real
  - e_i's are orthogonal

Eigenvectors & eigenvalues

- $e*M = \lambda*e$
- $e*M - \lambda*e = 0$
- $M*e - \lambda*e*I = 0$
- $M*e - \lambda*I*e = 0$
- $(M - \lambda*I) * e = 0$
  - e is ⊥ to row/columns of $(M - \lambda*I)$
- $\det(M - \lambda*I) = 0$
  - Polynomial => roots = eigenvalues

Hessian

Matrix H(f) of 2nd partial derivates

$$\begin{bmatrix} \frac{\partial f^2}{\partial x_0^2} & \frac{\partial f^2}{\partial x_0 \partial x_1} & \cdots & \frac{\partial f^2}{\partial x_0 \partial x_{n-1}} \\ \frac{\partial f^2}{\partial x_1 \partial x_0} & \frac{\partial f^2}{\partial x_1^2} & \cdots & \frac{\partial f^2}{\partial x_1 \partial x_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f^2}{\partial x_{n-1} \partial x_0} & \frac{\partial f^2}{\partial x_{n-1} \partial x_1} & \cdots & \frac{\partial f^2}{\partial x_{n-1}^2} \end{bmatrix}$$

$$f(\vec{x}) \text{ is } C^2 \text{ on } O(\vec{x}) \Rightarrow \frac{\partial f^2(\vec{x})}{\partial x_a \partial x_b} = \frac{\partial f^2(\vec{x})}{\partial x_b \partial x_a} \Rightarrow \text{Hessian is symmetric}$$

We are interested in 3D case:

$$\begin{bmatrix} \frac{\partial f^2}{\partial x^2} & \frac{\partial f^2}{\partial x \partial y} & \frac{\partial f^2}{\partial x \partial z} \\ \frac{\partial f^2}{\partial y \partial x} & \frac{\partial f^2}{\partial y^2} & \frac{\partial f^2}{\partial y \partial z} \\ \frac{\partial f^2}{\partial z \partial x} & \frac{\partial f^2}{\partial z \partial y} & \frac{\partial f^2}{\partial z^2} \end{bmatrix}$$

Computing Hessian

- Central differences of central differences
- 2nd derivation of reconstruction filter
  - Filter is at least at least C^2

Eigenvectors of a Hessian

$\vec{v} \cdot H(f) \cdot \vec{v}^T = 2^{nd}$ derivation in direction of $\vec{v}$ where $\|\vec{v}\| = 1$

- Hessian is real and symmetric
  - Eigenvalues are real
  - Eigenvertors are real and orthogonal

$$max(\vec{v} \cdot H(f) \cdot \vec{v}^T ; \forall \vec{v}) = \lambda_0$$
$$min(\vec{v} \cdot H(f) \cdot \vec{v}^T ; \forall \vec{v}) = \lambda_{n-1}$$
$$\vec{e}_0 \cdot H(f) \cdot \vec{e}_0^{\,T} = \lambda_0$$
$$\vec{e}_{n-1} \cdot H(f) \cdot \vec{e}_{n-1}^{\,T} = \lambda_{n-1}$$

Computing eigenvalues and eigenvectors

- Various iterative methods
  - GSL real symmetric matrix
    - symmetric bidiagonalization & QR reduction
- For 3x3 symmetric matrix
  - Iterative methods are slow/inaccurate
  - Direct method
    - 1. find eigenvalues as roots of a polynomial
    - 2. compute eigenvectors

1. Find eigenvalues

- $e*H = \lambda*e \Rightarrow (H - \lambda*I)*e = 0$
- $\det(H - \lambda*I) = 0$
  - For 3x3 matrix, det is polynomial of 3rd degree
  - Eigenvalues = roots of polynomial
    - sin, cos, atan2, sqrt, 1/x^3, …
  - $1.\ \lambda_0 > \lambda_1 > \lambda_2$
  - $2.\ \lambda_0 > \lambda_1 = \lambda_2$
  - $3.\ \lambda_0 = \lambda_1 = \lambda_2$
  - ~ 5x faster then GSL on CPU with FP64

2. compute eigenvectors

- $e*H = \lambda*e \Rightarrow (H - \lambda*I)*e = 0$
- e is ⊥ to row/columns of $(H - \lambda*I)$
  - Rows r_0, r_1, r_2
    - At most 2 linearly independent rows !
      - Need rank$(H - \lambda*I)$
        - Elimination method
        - mul, div

2. compute eigenvectors

- two independent rows $\lambda_0 > \lambda_1 > \lambda_2$
  - Find rows r_a & r_b in $(H - \lambda_i{*}I)$
  - e_i = normalize(r_a x r_b)
    - e_i is ⊥ to all rows of $(H - \lambda_i{*}I)$!
- one independent row r $\lambda_0 > \lambda_1 = \lambda_2$
  - fixed e_0, variable e_1 & e_2
    - e_1 & e_2 lie in plane ⊥ to r
- $H - \lambda{*}I = 0$ $\lambda_0 = \lambda_1 = \lambda_2$
  - Any vector is eigenvector

Performance

- GSL
  - λ : e ~ 2.666 : 1
- Direct method
  - λ : e ~ 1 : 1.51

Numerical issues

$$\forall i : rank(H - \lambda_i I) = 2 \Leftrightarrow \lambda_0 > \lambda_1 > \lambda_2$$

$$\forall i : rank(H - \lambda_i I) = 0 \Leftrightarrow \lambda_0 = \lambda_1 = \lambda_2$$

$$rank(H - \lambda_0 I) = 2 \wedge rank(H - \lambda_{1,2} I) = 1 \Leftrightarrow \lambda_0 > \lambda_1 = \lambda_2$$

$$rank(H - \lambda_i I) + multiplicity\ of\ \lambda_i = 3$$

- Eigenvalues
  - Round-off errors may produce “distinct” eigenvalues !
- Eigenvectors
  - Elimination method may produce “independent” rows !

Numerical issues eigenvalues

- Hessian normalization
  - Find s=max(abs(H[i][j]))
  - Scale H by 1/s if s > 1.0
  - Compute eigenvalues (root finding)
  - Scale eigenvalues by s
- Use higher precision

Numerical issues
Eigenvectors

- $A = H - \lambda * I$
- Set rank=0
- Choose r and c : m=A[r][c]=max(abs(A[i][j]))
- If m~0 return rank
- Divide row r by m
- eliminate row r from A by column c
  - A[i] = A[i] – A[i][c] * A[r]
- Increment rank and choose new r and c...

Higher precision on FP32 limited HW

- FP32 – Single, FP64 – Double
- single_x2 – data type of 2x FP32
  - Use native FP32 support with additional native computations
  - Result = single_x2[0] + single_x2[1]
  - single_x2 in not Double !
- Operator x
  - $A \times B \neq A \times_{single} B$
  - $err(A \times B) = A \times B - A \times_{single} B$

Higher precision on FP32 limited HW

- Sum(A , B) |A| >=|B|
  - $sum = A +_{single} B$
  - $err = B -_{single} (S -_{single} A)$
- Mul(A, B)
  - $mul = A *_{single} B$
  - A_x2 = split(A)   B_x2 = split(B)
  - $err = ((A\_x2[0] *_{single} B\_x2[0] -_{single} mul) +_{single} A\_x2[0] *_{single} B\_x2[1] +_{single} A\_x2[1] *_{single} B\_x2[0]) +_{single} A\_x2[1] *_{single} B\_x2[1]$