NOTES ON OLA INVERSION

JAVORNIK

The OLA inversions are based on the Multichannel Three-Dimensional SOLA Inversions of [Jackiewicz et al., 2012]. These notes begin with the simplest form of inversion, which is two-dimensional. Three-dimensional inversions follow, and finally, the full 3D inversion with multiple scatterers. Scatterers can be horizontal and vertical flow velocities, for example.

0.1. **Notation.** M measurements, each denoted by a or c.
α and β are scatterers or perturbations to a wave field, such as flow velocities.
$K_a^\alpha(\mathbf{r}, z)$ = effect of scatterer α on measurement, a.
$q^\alpha(\mathbf{r}, z)$ = 3D distribution of scatterer, α.
$\mathbf{x} = (\mathbf{r}, z) = (x, y, z)$.
N = Number of horizontal positions, $\mathbf{r}$.
Our Fourier transform is

$$f(\boldsymbol{k}) = \left(\frac{1}{2\pi}\right)^2 \sum_{\boldsymbol{r}} f(\boldsymbol{r}) e^{-i\boldsymbol{k}r} \tag{1}$$

Our inverse transform is

$$f(\boldsymbol{r}) = \left(\frac{2\pi}{N}\right)^2 \sum_{k_i} f(\boldsymbol{k}_i) e^{i\boldsymbol{k}_i r} \tag{2}$$

Assume $d_x = 1$, which means the grid spacing in real space is equal to one.

$$h_k = \frac{2\pi}{N} \tag{3}$$

1. 2D Inversions

Two-dimensional inversions are the first implementation:

- drop z-dependence; only 2D kernels(x,y)
- only one scatterer, $\alpha = 1$
- K, Kernel data
- Noise covariance, matrix Λ, is the identity matrix
- τ, map data
- σ_h, the horizontal full-width at half-maximum of the target function.
- target function $\mathcal{T}$.

At each wavenumber, k, the inversion finds a single scalar.

Date: September 17, 2012.

1.0.1. *Approach.* The forward equation calculates travel-time maps, $\delta\tau$ (dropping the α index, since there is only one type of kernel):

$$\delta\tau_a(r) = \int d^2r' K_a(r'-r)q(r') + n_a(r) \tag{4}$$

The inversion finds an estimate of the model

$$q_{inv} = w\delta\tau, \tag{5}$$

given kernel data (K), noise (n), and an averaging kernel, $\mathcal{K} = wK$.

$$q_{inv}(r) = \int d^2r' \mathcal{K}(r'-r)q(r') + \sum_i w_a(r_i - r)n_a(r_i) \tag{6}$$

The goal is to find a set of weights w that make the averaging kernel similar to a target function while controlling the amount of noise.

1.0.2. *Finding inversion weights.* The inversion finds the weights, w, by minimizing the cost function,

$$X(\mu) = \int d^2x[\mathcal{K}(x) - \mathcal{T}(x)]^2 + \mu \sum_{i,j,a,b} w_a(r_i)\Lambda_{ab}(r_i - r_j)w_b(r_j) \tag{7}$$

The first term is the misfit between the averaging kernel and the target function. The second term is the noise. μ is a regularization parameter and allows a trade-off between the allowed misfit and the amount of noise.

Jackiewicz defines a linear set of equations, solving for w and a Lagrange multiplier, λ.

The target function, $\mathcal{T}$, is

$$\mathcal{T}(x) = Cexp(\frac{-\|r\|^2}{2\sigma_h^2}) \tag{8}$$

1.0.3. *Algorithm.* For each wavenumber, k, calculate the weights, for each map, w_a, using:

$$h_k^4 N_x^2 \sum_a \tilde{A}_{ca}(\mathbf{k})\tilde{w}_a(\mathbf{k}) + \delta_{\mathbf{k},0}C_c\lambda = h_k^2\tilde{t}_c(\mathbf{k}), \forall c, \mathbf{k} \tag{9}$$

and

$$h_k^2 N_x^2 \sum_a C_a\tilde{w}_a(\mathbf{0}) = 1 \tag{10}$$

Substituting definition of h_k, the equations become

$$(2\pi)^2 h_k^2 \sum_a \tilde{A}_{ca}(\mathbf{k})\tilde{w}_a(\mathbf{k}) + \delta_{\mathbf{k},0}C_c\lambda = h_k^2\tilde{t}_c(\mathbf{k}), \forall c, \mathbf{k} \tag{11}$$

and

$$(2\pi)^2 \sum_a C_a\tilde{w}_a(\mathbf{0}) = 1 \tag{12}$$

Matrix $\tilde{A}(k)$ has c rows and a columns.

- Calculate A in frequency space:

$$\tilde{A}_{ca}(\mathbf{k}) = (2\pi)^2 \tilde{K}_c^*(\mathbf{k})\tilde{K}_a(\mathbf{k}) + \mu\tilde{\Lambda}_{ca}(\mathbf{k}) \tag{13}$$

Here, c and a are different measurements. Then, $K_c(k)$ is row c, in the map-depth matrix.

Construct $\tilde{A}(\mathbf{k})$:

$$\tilde{A}(\mathbf{k}) = (2\pi)^2 \tilde{K}^*(\mathbf{k})\tilde{K}^T(\mathbf{k}) + \mu\tilde{\Lambda}(\mathbf{k}) \tag{14}$$

$\tilde{A}(\mathbf{k})$ has dimensions $M \times M$ and is constructed in both cases ($\mathbf{k} = \mathbf{0}$) and ($\mathbf{k} \neq \mathbf{0}$).

- If ($\mathbf{k} = \mathbf{0}$), Calculate C in real space:

$$C_c = \sum_x \sum_y K_c(x, y) \tag{15}$$

As long as the $(k_x = 0, k_y = 0)$ element of the Fourier transformed kernel data is the sum of the magnitude of the real-space matrix elements, then, we can use this equation for C:

$$C = \tilde{K}(k_x = 0, k_y = 0) \tag{16}$$

Dimensions of C submatrix are $M \times 1$.

- Calculate t

$$\tilde{t}_c(\mathbf{k}) = (2\pi)^2 \tilde{K}_c^*(\mathbf{k})\tilde{T}(\mathbf{k}) \tag{17}$$

for $k = 0$ and $k \neq 0$

Q: Why are there different equations based on k?

$$\mathbf{k} = (k_x, k_y) \Rightarrow k_x = k_y = 0 \tag{18}$$

Because $\mathbf{k} = 0$ is a special case and enforces the constraint of averaging kernels integrating to unity (for scatterer of interest) and zero (for other scatterers).

A, t, and C are elements of a matrix, B, which is part of a set of linear equations, $Bw = t$. Solve for $[w, \lambda]$. Dimensions of w are M x 1.

Construct matrix

$$\begin{bmatrix} (2\pi)^2 h_k^2 \tilde{A}(0) & C \\ C^T & 0 \end{bmatrix} \begin{bmatrix} \tilde{w}(0) \\ \lambda \end{bmatrix} = \begin{bmatrix} h_k^2 \tilde{t}(0) \\ 1/(2\pi)^2 \end{bmatrix}$$

Calculate q and solve for averaging kernel and noise.

1.0.4. *Finding q.* Once the weights, w_a, are found, we can find $q = \sum_a w_a \delta\tau_a$. q is the model, or estimate of the flow velocities.

$$\tilde{q}(\mathbf{k}) = (2\pi)^2 \tilde{w}^*(\mathbf{k})\tilde{\delta\tau}(\mathbf{k}) \tag{19}$$

$$\tilde{q}(\mathbf{k}) \rightarrow^{IFFT} q(\mathbf{r}) \tag{20}$$

1.0.5. *Calculate Averaging Kernel.* Construct averaging kernel, $\mathcal{K} = wK$.

$$\mathcal{K}(\mathbf{x}) = h_k^4 N_x^2 \sum_a \sum_k \tilde{w}_a(\mathbf{k}) \tilde{K}_a(\mathbf{k}) e^{i\mathbf{kr}} \tag{21}$$

Substituting $h_k = \frac{2\pi}{N_x}$

$$\mathcal{K}(\mathbf{x}) = (2\pi)^2 h_k^2 \sum_a \sum_k \tilde{w}_a(\mathbf{k}) \tilde{K}_a(\mathbf{k}) e^{i\mathbf{kr}} \tag{22}$$

Performing calculation in frequency or wave space ...

$$\tilde{\mathcal{K}}(\mathbf{k}) = (2\pi)^2 \sum_a \sum_k \tilde{w}_a(\mathbf{k}) \tilde{K}_a(\mathbf{k}) \tag{23}$$

$$\tilde{\mathcal{K}}(\mathbf{k}) \rightarrow^{IFFT} \mathcal{K}(\mathbf{x}) \tag{24}$$

$\mathcal{K}(\mathbf{x})$ has dimensions $N_x \times N_y \times 1 \times 1$.

1.0.6. *Calculate Noise.* Construct the noise covariance using the weights and input covariance, Λ.

$$\sigma^2 = h_k^6 N_x^4 \sum_{a,b} \sum_k \tilde{w}_a^*(\mathbf{k}) \tilde{\Lambda}_{ab}(\mathbf{k}) \tilde{w}_b(\mathbf{k}) \tag{25}$$

Substituting $h_k = \frac{2\pi}{N_x}$ and performing calculation in frequency (wave) space, The noise covariance matrix is

$$\tilde{Cov}(\mathbf{k}) = (2\pi)^4 \sum_{a,b} \tilde{w}_a^*(\mathbf{k}) \tilde{\Lambda}_{ab}(\mathbf{k}) \tilde{w}_b(\mathbf{k}) \tag{26}$$

$$\tilde{Cov}(\mathbf{k}) \rightarrow^{IFFT} Cov(\mathbf{x}) \tag{27}$$

The noise covariance matrix has dimensions $N_x \times N_y \times 1 \times 1$.

2. 3D Inversions

Second implementation:

- include z-dependence. Now, we can have 3D kernels(x,y,z)
- only one scatterer, $\alpha = 1$
- K, Kernel data
- Noise covariance, matrix Λ, is the identity matrix
- τ, map data
- σ_h, the horizontal full-width at half-maximum of the target function.
- target function $\mathcal{T}$.

There is no alpha index, because there is only one type of kernel. This is the 3d_scalar version.

2.0.7. *Approach.* Equation from Jackiewicz with z-dependence is:

$$\delta\tau_a(r) = \int d^2r' dz K_a(r' - r, z) q(r', z) + n_a(r) \tag{28}$$

For OLA inversion, first find q_{inv}, given $\delta\tau$ (measurement data), K (kernel data), and n (noise) by constructing averaging kernel.

$$q_{inv}(r; z_0) = \int d^2r' dz \mathcal{K}(r' - r, z; z_0) q(r', z) + \sum_i w_a(r_i - r; z_0) n_a(r_i) \tag{29}$$

Once the weights, w, are found, we can find $q = w\delta\tau$.

2.0.8. *Finding inversion weights.* Find weights, w, by minimizing the cost function, X,

$$X(\mu) = \int d^3x [\mathcal{K}(x; z_0) - \mathcal{T}(x; z_0)]^2 + \mu \sum_{i,j,a,b} w_a(r_i; z_0) \Lambda_{ab}(r_i - r_j) w_b(r_j; z_0) \tag{30}$$

The first term is the misfit, the second term is the noise. Question: Is μ like λ in RLS approach? Yes, they are both regularization parameters that help control the amount of noise versus the misfit between the averaging kernel and the target function.

The target function, $\mathcal{T}$, is

$$\mathcal{T}(x; z_0) = Cexp(\frac{-\|r\|^2}{2\sigma_h^2}) \times f(z; z_0) \tag{31}$$

The target function is a 2D Gaussian, peaked at $\mathbf{r} = \mathbf{0}$, multiplied by a 1D function of the vertical coordinate. The target is possibly peaked at depth $z = z_0$.

$$f(z; z_0) = \sum f_i \phi_i(z) \tag{32}$$

f_i is an input FITS file containing a matrix of dimensions $N_z \times 1$. We don't directly calculate $f(z; z_0)$ because we don't know the individual $\phi_i(z)$ values. $f(z; z_0)$ falls out of the calculation for t (see equation (33)). Then, Gaussian (1×1) times $f(z)$ produces a map-depth matrix of $1 \times N_z$ at every k_x, k_y.

(What does this 1D function look like? Can it just be a constant for now? For testing, set $f(z) = 1$ for a single z, or for all z. During testing $f(z)$, is a function peaked at some point, z_0.)

The inversion software calculates the value of C to ensure the target function integrates to unity. (Should entire Target function be input file? Yes, this feature was added recently. There are two options for the target function: f_i or the entire target function $\mathcal{T}$ as an input FITS file.) The target function has dimensions $N_x \times N_y \times 1 \times N_z$.

2.0.9. *Algorithm.* The cost function can be written as a set of linear equations. An inversion solves for w and the Lagrange multipliers, λ using:

$$h_k^4 N_x^2 \sum_a \tilde{A}_{ca}(\mathbf{k}) \tilde{w}_a(\mathbf{k}) + \delta_{\mathbf{k},0} C_c \lambda = h_k^2 \tilde{t}_c(\mathbf{k}), \forall c, \mathbf{k} \tag{33}$$

and

$$h_k^2 N_x^2 \sum_a C_a \tilde{w}_a(\mathbf{0}) = 1 \tag{34}$$

Substituting definition of h_k the equations become

$$(2\pi)^2 h_k^2 \sum_a \tilde{A}_{ca}(\mathbf{k}) \tilde{w}_a(\mathbf{k}) + \delta_{\mathbf{k},0} C_c \lambda = h_k^2 \tilde{t}_c(\mathbf{k}), \forall c, \mathbf{k} \tag{35}$$

and

$$(2\pi)^2 \sum_a C_a \tilde{w}_a(\mathbf{0}) = 1 \tag{36}$$

Matrix $\tilde{A}(k)$ has c rows and a columns.

- Calculate A in frequency space:

$$\tilde{A}_{ca}(\mathbf{k}) = (2\pi)^2 \int dz \tilde{K}_c^*(\mathbf{k}, z) \tilde{K}_a(\mathbf{k}, z) + \mu \tilde{\Lambda}_{ca}(\mathbf{k}) \tag{37}$$

Here, c and a are different measurements. Then, $K_c(k)$ is row c, in the map-depth matrix. The integral becomes

$$\int \left[\sum_{i'} K_{ii'} \phi_{i'}(z) \right] \left[\sum_{j'} K_{jj'} \phi_{j'}(z) \right] dz = \sum_{i'j'} K_{ii'} K_{jj'} \int \phi_{i'}(z) \phi_{j'}(z) dz \tag{38}$$

The overlap matrix, Θ is

$$\int \phi_{i'}(z) \phi_{j'}(z) dz = \Theta_{i'j'} \tag{39}$$

Theta is an input file in FITS format and has dimension $N_z \times N_z$.

$$\int F(z) K(z) dz = \int \left[\sum_{i'} F_{i'} \phi_{i'}(z) \right] \left[\sum_{j'} K_{jj'} \phi_{j'}(z) \right] dz = \sum_{i'j'} F_{i'} \Theta_{i'j'} K_{j'} = F^T \Theta K \tag{40}$$

Construct $\tilde{A}(\mathbf{k})$:

$$\tilde{A}(\mathbf{k}) = (2\pi)^2 \tilde{K}^*(\mathbf{k}) \Theta \tilde{K}^T(\mathbf{k}) + \mu \tilde{\Lambda}(\mathbf{k}) \tag{41}$$

When $K(k)$ is extracted from MAP_DEPTH_MATRIX, K, the dimensions are $M \times N_z$. $\tilde{A}(\mathbf{k})$ has dimensions $M \times M$ and is constructed in both cases ($\mathbf{k} = \mathbf{0}$) and ($\mathbf{k} \neq \mathbf{0}$).

- If ($\mathbf{k} = \mathbf{0}$), Calculate C in real space:

$$C_c = \int K_c(x, y, z) \tag{42}$$

If

$$\left[1 = \sum_i \phi_i(z) \right] \tag{43}$$

$$\begin{aligned} C &= \int K(z) dz \\ &= \int K(z) \cdot 1 dz \\ &= \int K(z) \sum_i \phi_i(z) dz \\ &= \int \left[\sum_j K_j \phi_j(z) \right] \sum_i \phi_i(z) dz \\ &= K\Theta\mathbf{1} \end{aligned}$$

As long as the ($k_x = 0, k_y = 0, k_z = 0$) element of the Fourier transformed kernel data is the sum of the magnitude of the real-space matrix elements, then, we can use this equation for C:

$$C = \tilde{K}(k_x = 0, k_y = 0, k_z = 0) \tag{44}$$

K is $M \times N_z$, Theta is $N_z \times N_z$, then, we need something that is $N_z \times \mathbf{1}$. Dimension of C submatrix is $M \times 1$.

- Calculate t

$$\tilde{t_c}(\mathbf{k}) = (2\pi)^2 \int \tilde{K_c^*}(\mathbf{k}, z) \tilde{\mathcal{T}}(\mathbf{k}, z) dz = K^T \Theta \mathcal{T} \tag{45}$$

$K(k)$ has dimensions $M \times N_z$ and $\tilde{\mathcal{T}}(\mathbf{k})$ has dimensions $1 \times N_z$. $\tilde{t}(\mathbf{k})$ needs to have dimensions $M \times 1$. Then, we need to take the transpose of $\tilde{\mathcal{T}}(\mathbf{k}, z)$. In the code, we'll use this equation:

$$\tilde{t}(\mathbf{k}) = (2\pi)^2 \tilde{K}^*(\mathbf{k}) \Theta \tilde{\mathcal{T}}^T(\mathbf{k}) \tag{46}$$

for $k = 0$ and $k \neq 0$:

A, t, and C are elements of a matrix, B, which is part of a set of linear equations, $Bw = t$. Solve for [w, λ]. Dimensions of w are M x 1.

Construct matrix

$$\begin{bmatrix} (2\pi)^2 h_k^2 \tilde{A}(0) & C \\ C^T & 0 \end{bmatrix} \begin{bmatrix} \tilde{w}(0) \\ \lambda \end{bmatrix} = \begin{bmatrix} h_k^2 \tilde{t}(0) \\ 1/(2\pi)^2 \end{bmatrix}$$

Calculate q and solve for averaging kernel and noise.

2.0.10. *Finding q.* The equation for the model remains the same because there is no z-dependence.

$$\tilde{q}(\mathbf{k}) = (2\pi)^2 \tilde{w}^*(\mathbf{k}) \tilde{\delta\tau}(\mathbf{k}) \tag{47}$$

$$\tilde{q}(\mathbf{k}) \rightarrow^{IFFT} q(\mathbf{r}) \tag{48}$$

2.0.11. *Calculate Averaging Kernel.* Construct averaging kernel, $\mathcal{K} = wK$.

$$\mathcal{K}(\mathbf{x}) = h_k^4 N_x^2 \sum_a \sum_k \tilde{w}_a(\mathbf{k}) \tilde{K}_a(\mathbf{k}, z) e^{i\mathbf{kr}} \tag{49}$$

Substituting $h_k = \frac{2\pi}{N_x}$

$$\mathcal{K}(\mathbf{x}) = (2\pi)^2 h_k^2 \sum_a \sum_k \tilde{w}_a(\mathbf{k}) \tilde{K}_a(\mathbf{k}, z) e^{i\mathbf{kr}} \tag{50}$$

Performing calculation in frequency (wave) space ...

$$\tilde{\mathcal{K}}(\mathbf{k}) = (2\pi)^2 \sum_a \sum_k \tilde{w}_a(\mathbf{k}) \tilde{K}_a(\mathbf{k}) \tag{51}$$

$$\tilde{\mathcal{K}}(\mathbf{k}) \rightarrow^{IFFT} \mathcal{K}(\mathbf{x}) \tag{52}$$

Dimensions of $\tilde{\mathcal{K}}(\mathbf{k})$ are $1 \times N_z$. Dimensions of $\tilde{w}_a(\mathbf{k})$ are $M \times 1$. Dimensions of $\tilde{K}_a(\mathbf{k})$ are $M \times N_z$.

$\mathcal{K}(\mathbf{x})$ has dimensions $N_x \times N_y \times 1 \times N_z$. The spatial integral of an averaging kernel should be one.

2.0.12. *Calculate Noise.* The noise variance and covariance calculations remain the same because they have no z-dependence.

3. 3D Vector Inversions

Third implementation:

- multiple scatterers, $\alpha > 1$
- K, Kernel data; three kernels for each map
- Noise covariance, matrix Λ, is the identity matrix
- τ, map data
- σ_h, the horizontal full-width at half-maximum of the target function.

The difference between RLS and OLA inversions for 3d_vector is that RLS can calculate all v_x, v_y, v_z at the same time. With OLA, the inversion is performed for one scatterer at a time, but using all the data, i.e. all the kernel files.

In the target function, $\mathcal{T}$, include the alpha index. This is the 3d_vector version. Because the OLA inversions are calculated separately for each scatterer (e.g. each v_x, v_y, v_z), three new problem dimensions (3d_vector_x, 3d_vector_y, and 3d_vector_z) distinguish the scatterer of interest.

3.0.13. *Approach.* Original equation from Jackiewicz is

$$\delta\tau_a(r) = \int_{\odot} d^2r'dz \sum_{\alpha} K_a^{\alpha}(r'-r,z)q^{\alpha}(r',z) + n_a(r) \tag{53}$$

For OLA inversion, first find q, given τ, K and n (noise) (Given $\delta\tau$ (measurement data), K (kernel data), and n (noise)) by constructing averaging kernel.

$$\begin{aligned} q^{\theta}_{inv}(r;z_0) &= \int_{\odot} d^2r'dz\mathcal{K}^{\theta}(r'-r,z;z_0)q^{\theta}(r',z) \\ &+ \int_{\odot} d^2r'dz \sum_{\alpha,\alpha\neq\theta} \mathcal{K}^{\alpha}(r'-r,z;z_0)q^{\alpha}(r',z) \\ &+ \sum_{i,\alpha} w_a(r_i-r;z_0)n_a(r_i) \end{aligned}$$

Construct averaging kernel, $\mathcal{K} = wK$. Indirectly. Actually, the averaging kernel definition is used in the equations to find w, when minimizing cost function X.

Once the weights, w, are found, we can find $q = w\delta\tau$.

3.0.14. *Finding inversion weights.* Find weights, w, by minimizing a cost function, X,

$$X(\mu) = \int_{\odot} d^3x \sum_{\alpha} [\mathcal{K}^{\alpha}(x;z_0) - \mathcal{T}^{\alpha}(x;z_0)]^2 + \mu \sum_{i,j,a,b} w_a(r_i;z_0)\Lambda_{ab}(r_i-r_j)w_b(r_j;z_0) \tag{54}$$

First term is the misfit, the second term is the noise.

These are linear set of equations, solving for w and λ, a Lagrange multiplier.

The target function changes slightly to include multiple scatterers.

$$\mathcal{T}^{\alpha}(x;z_0) = Cexp(\frac{-\|r\|^2}{2\sigma_h^2}) \times f(z;z_0)\delta_{\alpha,\theta} \tag{55}$$

The target function is a Gaussian, peaked at z_0. σ_h controls the width of the Gaussian. C is a constant to ensure that the spatial integral of the target is unity. All targets are zero except for the scattter of interest, $\alpha = \theta$. σ_h is Full Width at Half Maximum (FWHM). For a normal distribution, FWHM $= 2\sqrt{2ln2}\sigma$ or $\sigma_h \approx 2.354\sigma$. The input parameter is σ_h, then $\sigma = \frac{\sigma_h}{2\sqrt{2ln2}}$. With the factor $C = \frac{1}{\sqrt{2\pi\sigma^2}}$, the spatial integral of the target function is unity. The target function is a 2D Gaussian, peaked at $\mathbf{r} = \mathbf{0}$, multiplied by a 1D function of the vertical coordinate. The target is possibly peaked at depth $z = z_0$.

$$f(z;z_0) = \sum f_i\phi_i(z) \tag{56}$$

f_i is an input FITS file which contains a matrix of dimensions $(N_z \times 1)$. The inversion code does not directly calculate $f(z;z_0)$ because the individual $\phi_i(z)$ values are not known. $f(z;z_0)$ falls out of the calculation for $\mathcal{T}$ and is not needed. Gaussian (1×1) times $f(z)$ produces a map-depth matrix of $1 \times (N_z \times \alpha)$.

Three dimensional vector inversions use the same $f_i(z)$ input file as 3D scalar inversion; the dimension does not increase. Then the target function for each scatterer is zero except for the one of interest. For example, with problem dimension 3d_vector_y, and flow velocities v_x, v_y, v_z for scatterers, the target functions, $\mathcal{T}^{v_x} = 0, \mathcal{T}^{v_z} = 0$, and $\mathcal{T}^{v_y}$ is the Gaussian described in (55).

3.0.15. *Algorithm.* For each wave number, k, calculate the weights, w_a, using:

$$h_k^4 N_x^2 \sum_a \tilde{A}_{ca}(\mathbf{k})\tilde{w}_a(\mathbf{k}) + \delta_{\mathbf{k},0} \sum_\alpha C_c^\alpha \lambda^\alpha = h_k^2 \tilde{t}_c(\mathbf{k}), \forall c, \mathbf{k} \tag{57}$$

and

$$h_k^2 N_x^2 \sum_a C_a^\alpha \tilde{w}_a(\mathbf{0}) = \delta_{\alpha,\theta}, \forall \alpha \tag{58}$$

Substituting definition of h_k the equations become

$$(2\pi)^2 h_k^2 \sum_a \tilde{A}_{ca}(\mathbf{k})\tilde{w}_a(\mathbf{k}) + \delta_{\mathbf{k},0} \sum_\alpha C_c^\alpha \lambda^\alpha = h_k^2 \tilde{t}_c(\mathbf{k}), \forall c, \mathbf{k} \tag{59}$$

and

$$(2\pi)^2 \sum_a C_a^\alpha \tilde{w}_a(\mathbf{0}) = \delta_{\alpha,\theta}, \forall \alpha \tag{60}$$

Matrix $\tilde{A}(k)$ has c rows and a columns.

- Calculate A in frequency space:

$$\tilde{A}_{ca}(\mathbf{k}) = (2\pi)^2 \int dz \sum_\alpha \tilde{K}_c^{\alpha *}(\mathbf{k}, z)\tilde{K}_a^\alpha(\mathbf{k}, z) + \mu\tilde{\Lambda}_{ca}(\mathbf{k}) \tag{61}$$

Here, c and a are different measurements. Then, $K_c(k)$ is row c, in the map-depth matrix.

The integral becomes

$$\int \left[\sum_{i'} K_{ii'}\phi_{i'}(z)\right]\left[\sum_{j'} K_{jj'}\phi_{j'}(z)\right] dz = \sum_{i'j'} K_{ii'}K_{jj'} \int \phi_{i'}(z)\phi_{j'}(z)dz \tag{62}$$

The overlap matrix, Θ is

$$\int \phi_{i'}(z)\phi_{j'}(z)dz = \Theta_{i'j'} \tag{63}$$

Theta is an input file in FITS format and has dimension $(N_z \times \alpha) \times (N_z \times \alpha)$.

$$\int F(z)K(z)dz = \int \left[\sum_{i'} F_{i'}\phi_{i'}(z)\right]\left[\sum_{j'} K_{jj'}\phi_{j'}(z)\right] dz = \sum_{i'j'} F_{i'}\Theta_{i'j'}K_{j'} = F^T\Theta K \tag{64}$$

Construct $\tilde{A}(\mathbf{k})$:

$$\tilde{A}(\mathbf{k}) = (2\pi)^2 \sum_{\alpha} \tilde{K}^{\alpha *}(\mathbf{k}) \Theta \tilde{K}^{\alpha T}(\mathbf{k}) + \mu \tilde{\Lambda}(\mathbf{k}) \tag{65}$$

Consider keeping matrices as (α, M). Keeping α as the rows, then we don't need to transpose.

$$\begin{aligned} A_{ca} &= \sum_{\alpha} K^*_{\alpha,c} K_{\alpha,a} \\ A &= K^H \Theta K \end{aligned}$$

When $K(k)$ is extracted from MAP_DEPTH_MATRIX, K, the dimensions are $M \times (N_z \times \alpha)$. $\tilde{A}(\mathbf{k})$ has dimensions $M \times M$ and is constructed in both cases $(\mathbf{k} = \mathbf{0})$ and $(\mathbf{k} \neq \mathbf{0})$.

- If $(\mathbf{k} = \mathbf{0})$, Calculate C in real space:

$$C = C^{\alpha}_{a} = \int_{\odot} K^{\alpha}_{a}(\boldsymbol{x}) d^3\boldsymbol{x} \tag{66}$$

If

$$\left[1 = \sum_i \phi_i(z) \right] \tag{67}$$

$$\begin{aligned} C &= \int K(z) dz \\ &= \int K(z) \cdot 1 dz \\ &= \int K(z) \sum_i \phi_i(z) dz \\ &= \int \left[\sum_j K_j \phi_j(z) \right] \sum_i \phi_i(z) dz \\ &= K \Theta \mathbf{1} \end{aligned}$$

$$
\begin{aligned}
C &= \int d^3\boldsymbol{x} K(\boldsymbol{x}, z) \\
K(\boldsymbol{x}, z) &= \left(\frac{2\pi}{N}\right)^2 \sum_{k_i} K(\boldsymbol{k}_i, z) e^{i\boldsymbol{k}_i x} \\
C &= \left(\frac{2\pi}{N}\right)^2 \int d^3\boldsymbol{x} \sum_{k_i} K(\boldsymbol{k}_i, z) e^{i\boldsymbol{k}_i x} \\
&= \left(\frac{2\pi}{N}\right)^2 (\Delta x)^2 \sum_j \sum_i \int dz \sum_{k_i} K(\boldsymbol{k}_i, z) e^{i\boldsymbol{k}_i x} \\
\sum_j e^{i\boldsymbol{k}_i x} &= \begin{cases} N^2 & k_i = 0 \\ 0 & else \end{cases} \\
C &= (2\pi)^2 (\Delta x)^2 \int dz K(k_i = 0, z)
\end{aligned}
$$

With $(\Delta x)^2 = 1$, we can use the $k_i = 0$ element of the kernel and multiply it by $(2\pi)^2$ to get the spatial integral of the kernel.

K is $M \times (N_z \times \alpha)$, Theta is $(N_z \times \alpha) \times (N_z \times \alpha)$, then, we need something that is $(N_z \times \alpha) \times \alpha$. Dimension of C submatrix is $M \times \alpha$. For 3d_vector, since $\delta_{\alpha,\theta}$ switches between one and zero depending on the α of interest, we need to have the different values present in the matrix solution for $k = 0$. $\delta_{\alpha,\theta}$ will have three values instead of one (as in the 3d_scalar case). Calculate C separately for each α. Need to separate C^α from each other. C^α is the spatial integral of scatterer, α. A special form of $\mathbf{1}$ could be ...

$$
\begin{array}{ccc}
C^{\alpha=1} & C^{\alpha=2} & C^{\alpha=3} \\
\begin{pmatrix}
1 & 0 & 0 \\
\vdots & \vdots & \vdots \\
1 & 0 & 0 \\
0 & 1 & 0 \\
\vdots & \vdots & \vdots \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\vdots & \vdots & \vdots \\
0 & 0 & 1
\end{pmatrix}
\end{array}
$$

with dimension $(N_z \times \alpha) \times \alpha$.

- Calculate t

$$
\tilde{t}_c(\mathbf{k}) = (2\pi)^2 \int \tilde{K}_c^{\theta *}(\mathbf{k}, z) \tilde{\mathcal{T}}^\theta(\mathbf{k}, z) dz = K^T \Theta \mathcal{T} \tag{68}
$$

$K(k)$ has dimensions $M \times (N_z \times \alpha)$ and $\tilde{\mathcal{T}}(\mathbf{k})$ has dimensions $1 \times (N_z \times \alpha)$. $\tilde{t}(\mathbf{k})$ needs to have dimensions $M \times 1$. Then, we need to take the transpose of $\tilde{\mathcal{T}}(\mathbf{k}, z)$. In the code, we'll use this equation:

$$\tilde{t}(\mathbf{k}) = (2\pi)^2 \tilde{K}^*(\mathbf{k}) \Theta \tilde{\mathcal{T}}^T(\mathbf{k}) \tag{69}$$

Remember, try to store matrices as (α, c) to avoid transpose of $\mathcal{T}$. $\tilde{\mathcal{T}}^T(\mathbf{k})$ could be ...

$$\begin{bmatrix} \mathcal{T}^{\alpha=1} \\ \mathcal{T}^{\alpha=2} \\ \mathcal{T}^{\alpha=3} \end{bmatrix}$$

All elements will be zero except where $\alpha = \theta$ (the scatterer of interest).

Linear Equation Solve. For $k = 0$ and $k \neq 0$: A, t, and C are elements of a matrix, B, which is part of a set of linear equations, $Bw = t$. Solve for [w, λ]. Dimensions of w are M x 1.

Construct matrix

$$\begin{bmatrix} (2\pi)^2 h_k^2 \tilde{A}(0) & C^{\alpha=1} & C^{\alpha=2} & C^{\alpha=3} \\ C^T_{\alpha=1} & 0 & 0 & 0 \\ C^T_{\alpha=2} & 0 & 0 & 0 \\ C^T_{\alpha=3} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{w}(0) \\ \lambda^{\alpha=1} \\ \lambda^{\alpha=2} \\ \lambda^{\alpha=3} \end{bmatrix} = \begin{bmatrix} h_k^2 \tilde{t}(0) \\ \delta_{\alpha=1,\theta}/(2\pi)^2 \\ \delta_{\alpha=2,\theta}/(2\pi)^2 \\ \delta_{\alpha=3,\theta}/(2\pi)^2 \end{bmatrix}$$

Calculate q and solve for averaging kernel and noise.

3.0.16. *Finding q.* With multiple scatterers, q does not change dimensions. Given the weights, w_a, the model is $q = \sum_a w_a \delta\tau_a$.

$$\tilde{q}(\boldsymbol{k}) = (2\pi)^2 \tilde{w}^*(\boldsymbol{k}) \tilde{\delta\tau}(\boldsymbol{k}) \tag{70}$$

$$\tilde{q}(\boldsymbol{k}) \rightarrow^{IFFT} q(\boldsymbol{r}) \tag{71}$$

3.0.17. *Calculate Averaging Kernel.* Construct averaging kernel, $\mathcal{K} = wK$.

$$\mathcal{K}^\alpha(\boldsymbol{x}) = h_k^4 N_x^2 \sum_a \sum_k \tilde{w}_a(\boldsymbol{k}) \tilde{K}_a^\alpha(\boldsymbol{k}, z) e^{i\boldsymbol{kr}} \tag{72}$$

Substituting $h_k = \frac{2\pi}{N_x}$

$$\mathcal{K}^\alpha(\boldsymbol{x}) = (2\pi)^2 h_k^2 \sum_a \sum_k \tilde{w}_a(\boldsymbol{k}) \tilde{K}_a^\alpha(\boldsymbol{k}, z) e^{i\boldsymbol{kr}} \tag{73}$$

Performing calculation in frequency (wave) space ...

$$\tilde{\mathcal{K}}^\alpha(\boldsymbol{k}) = (2\pi)^2 \sum_a \sum_k \tilde{w}_a(\boldsymbol{k}) \tilde{K}_a^\alpha(\boldsymbol{k}) \tag{74}$$

$$\tilde{\mathcal{K}}^\alpha(\boldsymbol{k}) \rightarrow^{IFFT} \mathcal{K}^\alpha(\boldsymbol{x}) \tag{75}$$

Dimensions of $\tilde{\mathcal{K}}(\mathbf{k})$ are $1 \times N_z$. Dimensions of $\tilde{w}_a(\mathbf{k})$ are $M \times 1$. Dimensions of $\tilde{K}_a(\mathbf{k})$ are $M \times N_z$.

$\mathcal{K}(\mathbf{x})$ has dimensions $N_x \times N_y \times 1 \times N_z$. The spatial integral of an averaging kernel should be one.

3.0.18. *Calculate Noise.* Construct Noise term, Cov.

$$\sigma^2 = h_k^6 N_x^4 \sum_{a,b} \sum_k \tilde{w}_a^*(\mathbf{k}) \tilde{\Lambda}_{ab}(\mathbf{k}) \tilde{w}_b(\mathbf{k}) \tag{76}$$

Substituting $h_k = \frac{2\pi}{N_x}$ and performing calculation in frequency (wave) space,

$$\tilde{Cov}(\mathbf{k}) = (2\pi)^4 \sum_{a,b} \tilde{w}_a^*(\mathbf{k}) \tilde{\Lambda}_{ab}(\mathbf{k}) \tilde{w}_b(\mathbf{k}) \tag{77}$$

The noise variance (σ^2) is a single number and it is the value of the noise covariance at $(\delta x, \delta y) = 0$, the center point of the noise covariance matrix.

$$\tilde{Cov}(\mathbf{k}) \rightarrow^{IFFT} Cov(\mathbf{x}) \tag{78}$$

The noise covariance matrix has dimensions $N_x \times N_y \times 1 \times 1$.

3.0.19. *Calculate $d \sim$.* This calculation does not make sense in the 3d_vector case because we would need all three v_x, v_y, v_z pieces of information to reconstruct the map. This calculation could be performed as a separate task.

4. 3D Vector Plus Kernel Weight Vector (Experimental)

Fourth implementation:

- add a weight vector as input parameter to scale the individual K^α.

The input will be the same as in the OLA 3d_vector inversion with the addition of a Kernel weighting vector w_1, w_2, w_3:

- M (RLS) = K (OLA), Kernel data; three kernels for each map
- Noise covariance, matrix Λ, is the identity matrix
- τ, map data
- σ, the horizontal full-width at half-maximum of the target function.

$$w_1^2(K_1 - \mathcal{T}_1)^2 + w_2^2(K_2 - \mathcal{T}_2)^2 + w_3^2(K_3 - \mathcal{T}_3)^2 \tag{79}$$

which is the same as scaling the kernels and target function by the weights.

$$= (w_1 K_1 - w\mathcal{T}_1)^2 + (w_2 K_2 - w_2 \mathcal{T}_2)^2 + (w_3 K_3 - w_3 \mathcal{T}_3)^2 \tag{80}$$

Question: Are we using kernels K or averaging kernels $\mathcal{K}$ here? I think it is averaging kernels?

Appendix A. Appendix: Design and Implementation Decisions

A.1. **Requirements and Code Changes.** tdinvert is replaced by OLA. So, tdinvert should be RLSInvert. Then we can have OLAInvert. The matrix, linearEqSolve, fftw, read and write FITS can be reused. Convert these to libraries.

Requirements:

- separate executables for OLAInvert and RLSInvert.

Need to do:

- LambdaSet should really be Range then, lambda (RLS) and mu (OLA) are instances of Range.

A.2. **Input File.** Some design considerations: If we don't use '2D' for OLA inversion and infer either 2D or 3D from the z-dimension of the kernel file(s), then the same input file can be used for OLAInvert and RLSInvert. The Regularization matrix (RLS) or Overlap matrix (OLA) oops, they mean different things base on the type of inversion, so they cannot be the same input file. Plus, μ and σ mean different things than λ_1 and λ_2. So, there is no reason to try to keep the input files the same for RLS and OLA. The meaning of the inputs are different. The code for parsing the input files can be the same, however. The RLS/OLA code can then interpret the meaning of the parameters.

Parse the kernel-map pairs. Need to know if kernel-map pairs are:

- single kernel files (1D)
- kernel-map pairs (2D, 3D scalar)
- kernel1, kernel2, kernel3, map triples (3D vector)

Also, need to know where lambda parameters are located. Which line of input? Just parse the input file into lines, send these lines to RLS/OLA. Let RLS and OLA determine the meaning of the lines based on the problem dimension.

The python script, *invert*, verifies files and directories exist, copies input files, versions the output directory, and calls the appropriate executable.

- σ, the horizontal full-width at half-maximum of the target function (input parameter).
- μ, input parameter.

Parameter	OLA	RLS
dimension	2d, 3d-scalar, 3d-vector_x, _y, _z	1d, 3d-scalar, 3d-vector
output base directory	yes	yes
base dir for kernels	yes	yes
base dir for maps	yes	yes
kernel-map pairs	K and τ	M and d
noise covariance	Λ	C
Regularization matrix	no	R
Averaging Kernel and noise depth calculations	yes	yes
looping parameters	μ, σ	λ_1, λ_2
Full-width at half-maximum of 2D Gaussian for Target function	σ	no
overlap (3D scalar only)	θ	no
1D function of z, $f(z)$ for Target function (3D scalar only)	f	no

A.3. **Libraries.** Libraries or core should contain:

- matrix (uses fitsio object?)
- MapDepthMatrix (uses MATRIX object)
- linearEqSolve (uses MATRIX object)
- read and write FITS files (uses MATRIX, MAP_DEPTH_MATRIX, and fitsio objects)
- fftw (uses MATRIX and fitsio objects)
- Parser for input file (Parser)

Use shared libraries instead of static libraries, because shared libraries produce smaller executable files and the library can be updated without recompiling the programs as long as the interface to the libraries doesn't change. Place libraries in

`/usr/local/lib`

Typedefs and structs are not part of a library, these header files are separate and need to be placed in

`/usr/include/coramf`

Use $gcc - Idir$ to compile with shared library headers. Think about how to structure code for release with shared libraries.

Reference

`http://www.network-theory.co.uk/docs/gccintro/gccintro_25.html`

The simplest way to set the load path is through the environment variable

`LD_LIBRARY_PATH.`

Currently, Makefiles contain link command with

```
-Wl,-rpath,$(DEFAULT_LIB_INSTALL_PATH)
```

"During development, there's the potential problem of modifying a library that's also used by many other programs – and you don't want the other programs to use the 'developmental' library, only a particular application that you're testing against it. One link option you might use is ld's 'rpath' option, which specifies the runtime library search path of that particular program being compiled. From gcc, you can invoke the rpath option by specifying it this way:

```
-Wl,-rpath,$(DEFAULT_LIB_INSTALL_PATH)
```

If you use this option when building the library client program, you don't need to bother with LD_LIBRARY_PATH (described next) other than to ensure it's not conflicting, or using other techniques to hide the library." From

`http://tldp.org/HOWTO/Program-Library-HOWTO/shared-libraries.html`

References

J. Jackiewicz, A. C. Birch, L. Gizon, S. M. Hanasoge, T. Hohage, J.-B. Ruffio, and M. Švanda. Multichannel Three-Dimensional SOLA Inversion for Local Helioseismology. *Solar Physics*, 276:19–33, February 2012. doi: 10.1007/s11207-011-9873-8.