1 Poisson Editing: audios, images, videos

Study objectives:
▷ two basic types of differential encoding (representation) and decoding (reconstruction) of temporal, spatial or spatio-temporal data;
▷ differential models, the initial value problem and the boundary value problem, related to optimal matching,
▷ discretization and numerical solutions.

1.1 Audio editing: preliminary

Problem description. Provided with a host signal sequence \( y = \{y_j\} \) over a finite time interval \([a, b]\),

\[
y_j = y(t_j), \quad t_j = a + j \cdot h, \quad j = 0, 1, \cdots, n, \quad t_0 = a, \quad t_n = b,
\]

where \( h > 0 \) is a *finite* time step size\(^1\). There are \( n \) time steps, 2 temporal boundary points and \((n - 1)\) interior points. Let \( p = \{p_i\} \) be a shorter signal sequence of length \( n_p < n \). We attempt to blend a segment of \( y \), over a specified sub-region (editing site) \([c, d] \subset [a, b]\), with the patch signal \( p \), automatically and seamlessly.

A naive way is to cut and paste, replacing the \( y \)-segment over \([c, d]\) by \( p \). This approach would often result in visible seams, i.e., non-negligible discrepancy or discontinuity at and around the segment boundary \( \{c, d\} \). Specifically, the difference in intensity \(|y(t) - p(t)|\) and the discrepancy in local variations are non-negligible in small neighborhoods of the boundary points \( c \) and \( d \). We introduce an effective editing approach via differential encoding and decoding.

1.1.1 Differential encoding & decoding

Encoding/representation. Represent the sequence \( y \) by its finite difference sequence and one of its temporal boundary values – \( y(a) \) or \( y(b) \),

\[
(\delta y)_j = y_j - y_{j-1}, \quad 1 \leq j \leq n,
\]

\[
y_0 = y_a, \quad y_n = y_b \quad \text{temporal boundary values}
\]

Decoding/reconstruction. The sequence \( y \) of (1) can be recovered, losslessly (faithfully), from the finite-difference sequence \( \delta y \), starting with *either* one of the temporary boundary values as the initial value (IV),

\[
y_j = y_{j-1} + (\delta y)_j, \quad j = 1, 2, \cdots, n, \quad \text{reconstruction from } y(a),
\]

\[
y_{j-1} = y_j - (\delta y)_j, \quad j = n, \cdots, 2, 1, \quad \text{reconstruction from } y(b).
\]

Each reconstruction is an integration process, using only one of the temporal boundary values. In practice, the other one can be used for checking reconstruction error.

**Take-away message:** every signal sequence \( y \) of (1) can be represented, and determined, by its difference sequence \( \delta y \) and an initial value.

---

\(^1\)In numerical computation and digital signal processing, the adjective 'finite' refers to a definitive step size \( h > 0 \) instead of vanishing, infinitesimal step size \( h \to 0 \) as in differential calculus.
Fundamental benefits:

- Range shift with shape preserving. A shift by $s$ in the initial value results in a shifted construction $\{s + y_j\}$, the signal shape is preserved in $\delta y$.
- Range rescaling with relative shape preserving. Dilate or shrink the $y$-range by scaling $\delta y$, with the chosen initial value fixed.
- Reshaping/filtering. via manipulating $\delta y$, which includes denoising, smoothing, regulating and editing. We will focus on editing.

Additional advantages:

- Range reduction. The numerical range of $\delta y$ is often substantially smaller than that of $y$, if $y$ is of the same sign (such as nonnegative in intensity).
- Compression or extraction more of shape information. For instance, any constant or linear subsequence in $\delta y$ can be represented simply by a constant value followed by the length of the subsequence.
- Recursion. Apply the encoding-decoding schemes to $\delta y$, in one more than one ways. We shall see shortly a second-order encoding/decoding scheme.

Differential encoding blockwise: mitigating a drawback:

There is a drawback in signal reconstruction: the sequential dependence is deeper with a longer sequence, which also implies long-range propagation of numerical reconstruction errors. A block-wise encoding can reduce the dependence length. Divide a long signal sequence into multiple subsequences, not necessarily of the same length. Apply the coding schemes to each and every subsequence. There are different ways to divide a sequence. A subsequence may be a consecutive segment of $y$, but not necessarily. For instance, we can divide a sequence into $k$-interleaving subsequences, $k > 1$. Let $\ell$ index the subsequences, $0 \leq \ell < k$. The subsequence-$\ell$ is $\{y_{kj+\ell}\}$. The dependence length in reconstruction is reduced by a factor of $k$.

In the rest of the section, we consider the single-piece encoding and decoding.

1.2 Embedding into a continuous-time signal

We relate sequence $y = \{y_j\}$ to its continuous counterpart $y(t)$. This might seem a detour for 1D signal editing. In fact, it serves as a short-cut for editing higher-dimensional signals, such as 2D or 3D images. This connection may also help expand and enhance our analysis skills in modeling and model solutions between continuous/smoothed formulation and discrete/discretized formulation.

We embed $\{y_j \mid h\}$ into a continuous time signal $y(t)$ such that

$$y(t)_{j+1} = y(t_j) + \int_{t_j}^{t_{j+1}} y'(\tau)d\tau.$$  

The signal $y(t)$ can be unique determined by its derivative and an initial value,

$$\begin{cases} 
  y'(t), & t \in [a, b], \\
  y(a) = y_a, & \text{or, } y(b) = y_b, 
\end{cases}$$

Reconstruction by integration.

$$y(t) = \int_a^t y'(\tau)d\tau, \quad \text{or, } y(t) = \int_t^b y'(\tau)d\tau.$$
1.3 Seamless blending model: continuous & discrete

1.3.1 A constrained optimal fitting problem

We look for a smooth blending/fitting signal $f(t)$ with the following properties:

- guaranteed continuity with the host signal $y(t)$ at $\partial \Omega_{\text{edit}}$, and
- best shape matching to the patch signal $p(t)$ (least variation) over $\Omega_{\text{edit}}$.

The following least-squared (LS) blending model is intuitive and simple. We minimize the squared-mismatch-error in shape over $[c, d]$, subject to exact interpolation at the boundary:

$$f_* = \arg \min_{f \in C^2[c, d]} \|f' - p'\|_2^2$$

subject to

$$f(c) = y(c), \quad f(d) = y(d).$$

(5)

Both boundary values are used. There is an additional regulation/feasibility condition that $f$ be twice differential. This is to avoid spurious interpolation at the boundary and over-fitting over the interior of the editing site.

1.3.2 Model reformulation: the Euler-Lagrange equation

By the variational principle \(^2\), we get the Euler-Lagrange equation associated with the LS-blending optimization (5). Specifically, the zero-gradient equation $\nabla g = 0$ materializes as the 1D Poisson (differential) equation with the Dirichlet boundary condition (BC),

$$\begin{cases} 
\Delta f(t) = \frac{d^2 f(t)}{dt^2} = p''(t), & t \in (c, d) \\
\text{BC:} & f(c) = y(c), \quad f(d) = y(d),
\end{cases}$$

(6)

given the host signal $y(t)$ over $[a, b]$ and the patch signal $p(t)$ over $[c, d] \subset [a, b]$. In the context of (6), $\Delta$ denotes the second derivative, or the 1D Laplacian operator.

- By the model reformulation, the LS-matching between the first derivatives, $f'$ and $p'$, arrives at the exact matching between the second derivatives, $f''$ and $p''$, subject to the same boundary value conditions. This insight manifests an analytical advantage of the Euler-Lagrange approach for optimization over the iterative gradient descending/ascending approach.

- The LS-blending signal $f$ is uniquely determined. Its construction is by integration from $\Delta f$ and the boundary values.

1.3.3 The discretized 1D Poisson equation

The Poisson equation is discretized and coupled with the boundary condition as follows for the LS-blending sequence $f$ at the $n_i = n - 1$ interior points,

$$\Delta_h f(1 : n_i) = b(1 : n_i)$$

(7)

\(^2\)The principle is based on the same arguments as for the LS residual equation
with the right-hand side \(b\) determined by \(\Delta_h p\) and the boundary values,

\[
b(1 : n_i) := \Delta_h p(1 : n_i) - [y(a)e_1 + y(b)e_{n_i}]/h^2,
\]

(8)
given the host sequence \(y\) and the patch sequence \(p\). Here, \(\Delta_h\) denotes the discretized Laplacian operator with discretization step size \(h\),

\[
\Delta = \frac{d^2}{dt^2} \implies \Delta_h = \frac{1}{h^2} \begin{pmatrix}
-2 & 1 & & & \\
1 & -2 & 1 & & \\
& \ddots & \ddots & \ddots & \\
& & 1 & -2 & 1 \\
& & & 1 & -2
\end{pmatrix},
\]

(9)

System (7) is linear in \(f\), sparse, highly structured. It is nonsingular, but the condition number for reconstruction increases with \(n\).

We have introduced along the way a second-order differential sequence encoding scheme, the Poisson encoding scheme. It can be viewed as a combination of the two first-order encoding schemes. The numerical reconstruction is more involved, the Laplacian matrix is not triangular, i.e., the dependency in the solution elements is no longer one-way sequential.

### 1.4 Image editing: two or higher dimensions

The host image \(y(x_1, x_2)\) is provided over a 2D spatial domain \(\Omega\). The patch image \(p(x_1, x_2)\) is to be used over an editing site \(\Omega_{\text{edit}} \subset \Omega\). We extend the key steps in 1D signal editing to higher dimensions:

- set up the LS-blending model,
- convert the model to the Euler-Lagrange equation,
- discretize the equation,
- get numerical solution for the optimal-blending image,

**The LS-blending model:**

\[
f_* = \arg \min_{f \in C^2(\Omega_{\text{edit}})} \| \nabla f - \nabla p \|^2
\]

s.t.

\[
f(\partial \Omega_{\text{edit}}) = y(\partial \Omega_{\text{edit}})
\]

(10)

The optimal solution matches in spatial gradients to the patch signal and matches in intensity to the host signal at the editing site boundary.

**The Euler-Lagrange equation** in particular is the Poisson equation:

\[
\begin{cases}
\Delta f(x_1, x_2) = \frac{\partial^2 f}{\partial x_1^2} + \frac{\partial^2 f}{\partial x_2^2} = \Delta p(x_1, x_2), & (x_1, x_2) \in \Omega_{\text{edit}} \\
\text{BC: } f(\partial \Omega_{\text{edit}}) = y(\partial \Omega_{\text{edit}})
\end{cases}
\]

(11)

provided with the host image \(y\), the patch image \(p\) and a specification of \(\Omega_{\text{edit}}\) with boundary \(\partial \Omega_{\text{edit}}\). It says, the LS-matching to \(p\) in the gradient field actually has the same Laplacian as \(p\) over the editing site, subject to the exact matching in intensity to \(y\) at the site boundary. The Laplacian \(\Delta f\) is the divergence of the gradient \(\nabla f\).
**Spatial discretization.** Let the editing site be enclosed by a rectangle box, $\Omega_{\text{edit}} \subset [c_1, d_1] \times [c_2, d_2]$. The image pixel locations correspond to equi-spaced discretization in $x_1$ and in $x_2$ with step sizes $h_1$ and $h_2$, respectively. Let $n_i h_i = d_i - c_i$, $i = 1, 2$. Then, the discretized equation for the optimal-blending image $f$ as a 2D pixel array $f(:, :)$ is as follows,

$$\Delta h_1 f(1 : n_1, 1 : n_2) + f(1 : n_1, 1 : n_2) \Delta h_2 = b(1 : n_1, 1 : n_2). \quad (12)$$

The pixel-array $b(:, :)$ on the right hand side is set as follows,

- on the pixels in $\Omega_{\text{edit}}$ not adjacent to $\partial \Omega_{\text{edit}}$, $b(:, :)$ is equal to $\Delta p(:, :)$, shaping the blending image $f$;
- on the interior pixels adjacent to $\partial \Omega_{\text{edit}}$, $b$ is equal to $\Delta p$ combined with the accent boundary values, similar to that in (8);
- on the pixels outside $\Omega_{\text{edit}}$, $b$ is $\Delta y$, preserving the shape of the host image.

In short, $b$ divides the bounding box into three parts, guides the editing in and outside of the editing site $\Omega_{\text{edit}}$ by the respective divergence fields, and aligns $f$ with the host image $y$ at the boundary. This setting makes the model flexible to the size and shape of a bounding box. The bounding box and the right-hand-side $b$ can accomodate more than one editing sites, as in one of our demo cases.

**Numerical solution methods.** Equation (12) is linear in $f$, large (with $n_1 n_2$ unknowns), sparse (each unknown is coupled only with 4 others), and highly structured in its sparsity pattern. Mathematically, the solution exists uniquely. Numerically, the particular Poisson equation can be solved with a direct method or an iterative method. In fact, this equation often serves as a case-study model for investigating an iterative method or a direct method. In terms of the sensitivity to perturbation, the condition number increases with $n_1$ and $n_2$. A smaller bounding box is preferred for better efficiency and less sensitivity.

The above extends straightforwardly to editing 3D images on spatial voxels.

### 1.5 Video editing

We have a host video $y(t, x_1, x_2)$ and a patch video $p(t, x_1, x_2)$ for composition. A video data set may be viewed as a time sequence of 2D image frames. By this view, a simple heuristic approach is to make video editing frame by frame. The resulting video may have unwanted non-smooth temporal transition from frame to frame. A video may also be viewed as a 2D pixel array of time sequences. It is not hard to imagine that if we make sequence editing along each pixel thread, pixel by pixel, there is no guarantee to retain spatial smoothness in any spatial neighborhood. These crude results, however, may serve as initial guesses for iterative editing.

Thirdly, a video dataset might be mistaken as a 3D image. There is a fundamental difference. In developing 3D image blending models, each spatial variable is treated independent of the others. For video blending models, however, the spatial variables $x_1$ and $x_2$ are not independent of the time variable, $x_1 = x_1(t)$ and $x_2 = x_2(t)$.

It is an interesting and non-trivial exercise to make video-blending seamless and fast.