Tracking particles by passing messages between images

Michael Chertkov

Center for Nonlinear Studies & Theory Division, LANL

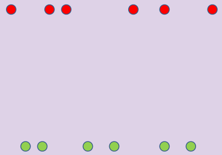
Joint work with
L. Kroc, F. Krzakala, M. Vergassola, L. Zdeborova

International Symposium on Turbulence
September 22, 2009 Peking University

Outline

One Problem and One Idea

The Problem



Standard Solution

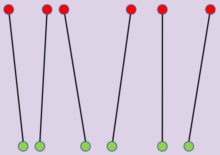
- Take snapshots often = Avoid trajectory overlap
- Consequence = A lot of data
- Gigabit/s to monitor a two-dimensional slice of a $10cm^3$ experimental cell with a pixel size of $0.1mm$ and exposition time of $1ms$
- Still need to "learn" velocity (diffusion) from matching

This Talk [suggestions to Eberhard, Victor, Jean-Francois, Charles ++]

- Take fewer snapshots = Let particles overlap
- Put extra efforts into Learning/Inference
- Use our (turbulence community) knowledge of Lagrangian evolution
- Focus on learning (rather than matching)

One Problem and One Idea

The Problem



Standard Solution

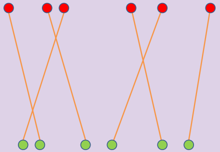
- Take snapshots often = Avoid trajectory overlap
- Consequence = A lot of data
- Gigabit/s to monitor a two-dimensional slice of a $10cm^3$ experimental cell with a pixel size of $0.1mm$ and exposition time of $1ms$
- Still need to "learn" velocity (diffusion) from matching

This Talk [suggestions to Eberhard, Victor, Jean-Francois, Charles ++]

- Take fewer snapshots = Let particles overlap
- Put extra efforts into Learning/Inference
- Use our (turbulence community) knowledge of Lagrangian evolution
- Focus on learning (rather than matching)

One Problem and One Idea

The Problem



Standard Solution

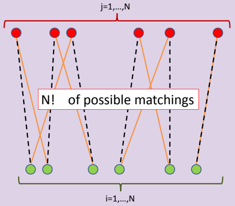
- Take snapshots often = Avoid trajectory overlap
- Consequence = A lot of data
- Gigabit/s to monitor a two-dimensional slice of a $10cm^3$ experimental cell with a pixel size of $0.1mm$ and exposition time of $1ms$
- Still need to "learn" velocity (diffusion) from matching

This Talk [suggestions to Eberhard, Victor, Jean-Francois, Charles ++]

- Take fewer snapshots = Let particles overlap
- Put extra efforts into Learning/Inference
- Use our (turbulence community) knowledge of Lagrangian evolution
- Focus on learning (rather than matching)

One Problem and One Idea

The Problem



And after all we actually don't need matching. Our goal is to LEARN THE FLOW.

Standard Solution

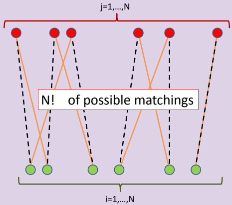
- Take snapshots often = Avoid trajectory overlap
- Consequence = A lot of data
- Gigabit/s to monitor a two-dimensional slice of a $10cm^3$ experimental cell with a pixel size of $0.1mm$ and exposition time of $1ms$
- Still need to "learn" velocity (diffusion) from matching

This Talk [suggestions to Eberhard, Victor, Jean-Francois, Charles ++]

- Take fewer snapshots = Let particles overlap
- Put extra efforts into Learning/Inference
- Use our (turbulence community) knowledge of Lagrangian evolution
- Focus on learning (rather than matching)

One Problem and One Idea

The Problem



And after all we actually don't need matching. Our goal is to LEARN THE FLOW.

Standard Solution

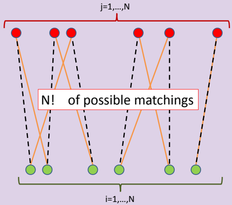
- Take snapshots often = Avoid trajectory overlap
- Consequence = A lot of data
- Gigabit/s to monitor a two-dimensional slice of a $10cm^3$ experimental cell with a pixel size of $0.1mm$ and exposition time of $1ms$
- Still need to "learn" velocity (diffusion) from matching

This Talk [suggestions to Eberhard, Victor, Jean-Francois, Charles ++]

- Take fewer snapshots = Let particles overlap
- Put extra efforts into Learning/Inference
- Use our (turbulence community) knowledge of Lagrangian evolution
- Focus on learning (rather than matching)

One Problem and One Idea

The Problem



And after all we actually don't need matching. Our goal is to LEARN THE FLOW.

Standard Solution

- Take snapshots often = Avoid trajectory overlap
- Consequence = A lot of data
- Gigabit/s to monitor a two-dimensional slice of a $10cm^3$ experimental cell with a pixel size of $0.1mm$ and exposition time of $1ms$
- Still need to "learn" velocity (diffusion) from matching

This Talk [suggestions to Eberhard, Victor, Jean-Francois, Charles ++]

- Take fewer snapshots = Let particles overlap
- Put extra efforts into Learning/Inference
- Use our (turbulence community) knowledge of Lagrangian evolution
- Focus on learning (rather than matching)

Lagrangian Dynamics under the Viscous Scale

Plausible (for PIV) Modeling Assumptions

- Particles are normally seed with mean separation few times smaller than the viscous scale.
- The Lagrangian velocity at these scales is spatially smooth.
- Moreover the velocity gradient, $\hat{s}$, at these scales and times is frozen (time independent).

Batchelor (diffusion + smooth advection) Model

- Trajectory of i's particles obeys: $d\mathbf{r}_i(t)/dt = \hat{s}\mathbf{r}_i(t) + \boldsymbol{\xi}_i(t)$
- $\mathrm{tr}(\hat{s}) = 0$ - incompressible flow
- $\langle \xi_i^\alpha(t_1)\xi_j^\beta(t_2)\rangle = \kappa\delta_{ij}\delta^{\alpha\beta}\delta(t_1 - t_2)$

Inference & Learning (I)

Main Task: Learning parameters of the flow and of the medium

- Given positions of N identical particles at $t=0$ and $t=1$: $\forall i,j=1,\cdots,N, \quad \mathbf{x}_i=\mathbf{r}_i(0)$ and $\mathbf{y}^j=\mathbf{r}_j(1)$
- To output MOST PROBABLE values of the flow, $\hat{s}$, and the medium, κ, characterizing the inter-snapshot span: $\boldsymbol{\theta}=(\hat{s};\kappa)$.

Sub-task: Inference [reconstruction] of Matchings

- Given parameters of the medium and the flow, $\boldsymbol{\theta}$
- To reconstruct Most Probable matching between identical particles in the two snapshots ["ground state"]
- Even more generally - Probabilistic Reconstruction: to assign probability to each matchings and evaluate marginal probabilities ["magnetizations"]

Inference & Learning (II)

All of the above — Formally

$$P_i^j(\mathbf{x}_i,\mathbf{y}^j) = (\det M)^{-\frac{1}{2}} \exp\left(-\tfrac{1}{2}\mathbf{r}^\alpha (M^{-1})^{\alpha\beta}\mathbf{r}^\beta\right)$$

$$\mathbf{r} = \mathbf{y}^j - W(\Delta)\,\mathbf{x}_i \quad W(t) = \exp(t\,\hat{s})$$

$$M = \kappa W(\Delta)\int_0^\Delta W^{-1}(t) W^{-1,T}(t)\mathrm{d}t\ W^T(\Delta)$$

Inference [of matchings]

- Maximum Likelihood: $\mathrm{argmax}_\sigma \mathcal{L}(\{\sigma\}|\boldsymbol{\theta})$
- Marginal Probability of a link $\binom{j}{i}$: $\sum_{\sigma\setminus\sigma_i^j}\mathcal{L}(\{\sigma\}|\boldsymbol{\theta})$

$$\mathcal{L}(\{\sigma\}|\boldsymbol{\theta}) = C(\{\sigma\})\prod_{(i,j)}\left[P_i^j\left(\mathbf{x}_i,\mathbf{y}^j|\boldsymbol{\theta}\right)\right]^{\sigma_i^j}, \quad C(\{\sigma\}) \equiv \prod_j \delta\left(\sum_i \sigma_i^j, 1\right)\prod_i \delta\left(\sum_j \sigma_i^j, 1\right)$$

Learning [of parameters]

- The best one can do: $\boldsymbol{\theta}^* = \mathrm{argmax}_{\boldsymbol{\theta}} Z(\boldsymbol{\theta})$

$$P(\boldsymbol{\theta}|\mathbf{x}_i,\mathbf{y}^j) \propto \sum_{\{\sigma\}}\mathcal{L}(\{\sigma\}|\boldsymbol{\theta}) \equiv Z(\boldsymbol{\theta})\ \text{- partition function}$$

Easy vs Difficult

- Difficult = Exponential in the system size, 2^N
- Easy [theory] = Polynomial in the system size
- **Really Easy** [can work "on the fly"] = Linear in the system size

- To find Maximum Likelihood Assignment is EASY
- To evaluate ALL other aforementioned tasks [Evaluation of the Partition function, Learning] are DIFFICULT

Belief Propagation is Heuristics of OUR choice

- Trades optimality for efficiency. **Really Easy.**
- Shows good performance in simulation test
- Has some pleasant theoretical guarantees

Belief Propagation (BP) and Message Passing

Bethe Free Energy formulation of BP [Yedidia, Freeman, Weiss '01]

Minimize the Kubblack-Leibler functional

$$\mathcal{F}\{b(\{\sigma\})\} \equiv \sum_{\{\sigma\}} b(\{\sigma\}) \ln \frac{b(\{\sigma\})}{\mathcal{L}(\{\sigma\})}$$

Difficult/Exact

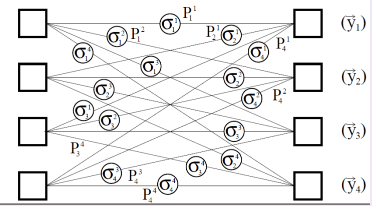
under the following "almost variational" substitution" for beliefs:

$$b(\{\sigma\}) \approx \frac{\prod_i b_i(\boldsymbol{\sigma}_i) \prod_j b^j(\boldsymbol{\sigma}^j)}{\prod_{(i,j)} b_i^j(\sigma_i^j)}$$

Easy/Approximate

- Message Passing is a Distributed Implementation of BP
- Graphical Models = the language

Tracking Particles as a Graphical Model



$$\mathcal{L}(\{\sigma\}|\boldsymbol{\theta}) = C\left(\{\sigma\}\right) \prod_{(i,j)} \left[P_i^j\left(\mathbf{x}_i, \mathbf{y}^j|\boldsymbol{\theta}\right)\right]^{\sigma_i^j}$$

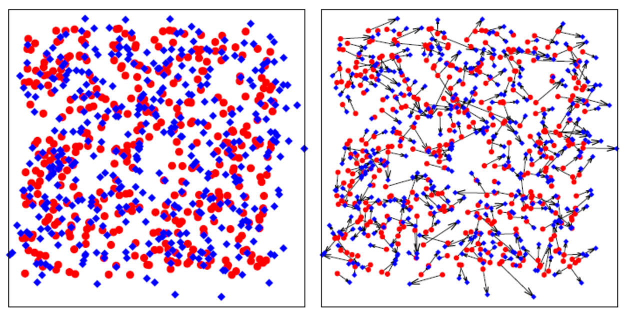
$$C\left(\{\sigma\}\right) \equiv \prod_j \delta\left(\sum_i \sigma_i^j, 1\right) \prod_i \delta\left(\sum_j \sigma_i^j, 1\right)$$

Surprising Exactness of BP for ML-assignement

- Exact Polynomial Algorithms (auction, Hungarian) are available for the problem
- Generally BP is exact only on a graph without loops [tree]
- In this [Perfect Matching on Bipartite Graph] case it is still exact in spite of many loops!! [Bayati, Shah, Sharma '08], also Linear Programming/TUM interpretation [MC '08]

Outline

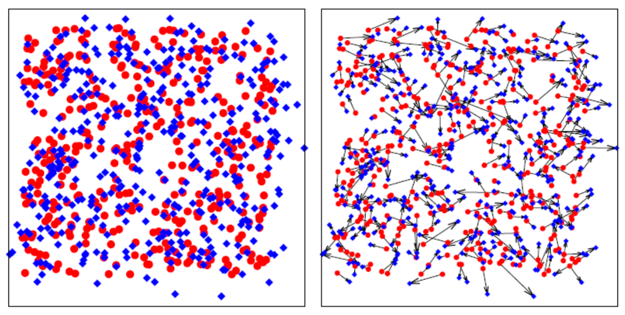
Can you guess who went where?



- N particles are placed uniformly at random in a d-dimensional box of size $N^{1/d}$
- Choose $\boldsymbol{\theta} = (\kappa, \mathbf{s})$ in such a way that after rescaling, $\hat{s}^* = \hat{s}N^{1/d}$, $\kappa^* = \kappa$, all the rescaled parameters are $O(1)$.
- Produce a stochastic map for the N particles from the original image to respective positions in the consecutive image.

- $N = 400$ particles. 2D.
- $\hat{s} = \begin{pmatrix} a & b-c \\ b+c & a \end{pmatrix}$
- Actual values: $\kappa = 1.05,\ a^* = 0.28,\ b^* = 0.54,\ c^* = 0.24$

Can you guess who went where?



- N particles are placed uniformly at random in a d-dimensional box of size $N^{1/d}$
- Choose $\boldsymbol{\theta} = (\kappa, \mathbf{s})$ in such a way that after rescaling, $\hat{s}^* = \hat{s}N^{1/d}$, $\kappa^* = \kappa$, all the rescaled parameters are $O(1)$.
- Produce a stochastic map for the N particles from the original image to respective positions in the consecutive image.

- $N = 400$ particles. 2D.
- $\hat{s} = \begin{pmatrix} a & b-c \\ b+c & a \end{pmatrix}$
- Actual values: $\kappa = 1.05$, $a^* = 0.28$, $b^* = 0.54$, $c^* = 0.24$
- Output of OUR LEARNING algorithm: [accounts for multiple matchings !!]
  $\kappa_{BP} = 1$, $a_{BP} = 0.32$, $b_{BP} = 0.55$, $c_{BP} = 0.19$ [within the "finite size" error]

Combined Message Passing with Parameters' Update

Fixed Point Equations for Messages

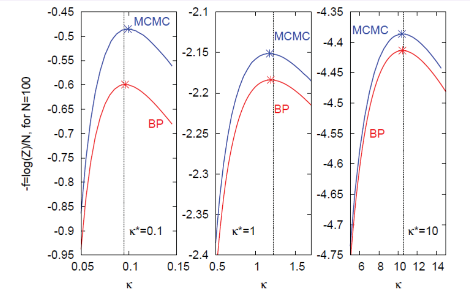
- BP equations: $\overline{h}^{i\to j} = -\frac{1}{\beta}\ln\sum_{k\neq j} P_i^k e^{\beta \underline{h}^{k\to i}}$; $\underline{h}^{j\to i} = -\frac{1}{\beta}\ln\sum_{k\neq i} P_k^j e^{\beta \overline{h}^{k\to j}}$
- BP estimation for $Z_{BP}(\boldsymbol{\theta}) = Z(\boldsymbol{\theta}|\mathbf{h}$ solves BP eqs. at $\beta = 1$)
- MPA estimation for $Z_{MPA}(\boldsymbol{\theta}) = Z(\boldsymbol{\theta}|\mathbf{h}$ solves BP eqs. at $\beta = \infty$)

$$Z(\boldsymbol{\theta}|\mathbf{h};\beta) = \sum_{(ij)} \ln\left(1 + P_i^j e^{\beta\overline{h}^{i\to j} + \beta\underline{h}^{j\to i}}\right) - \sum_i \ln\left(\sum_j P_i^j e^{\beta\underline{h}^{j\to i}}\right) - \sum_j \ln\left(\sum_i P_i^j e^{\beta\overline{h}^{i\to j}}\right)$$

Learning: $\text{argmin}_{\boldsymbol{\theta}} Z(\boldsymbol{\theta})$

- Solved using Newton's method in combination with message-passing: after each Newton step, we update the messages
- Even though (theoretically) the convergence is not guaranteed, the scheme always converges
- Complexity [in our implementation] is $O(N^2)$, even though reduction to $O(N)$ is straightforward

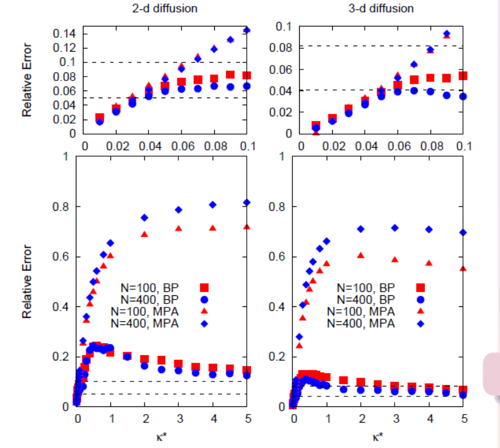
BP vs "Exact" Markov Chain Monte Carlo



- Fully Polynomial Randomized Scheme FPRS/MCMC "exact" is available [for estimating a permanent of a positive matrix Jerrum, Sinclair, Vigoda '04]
- Honest complexity of FPRS/MCMC is $O(N^{11})$ ... still $O(N^3)$ even after an acceleration
- MCMC is not distributed
- Accuracy of BP prediction (for maximum) is perfect!!
- Our (BP) scheme is significantly faster

BP vs MPA (I): Pure Diffusion. Relative Error.

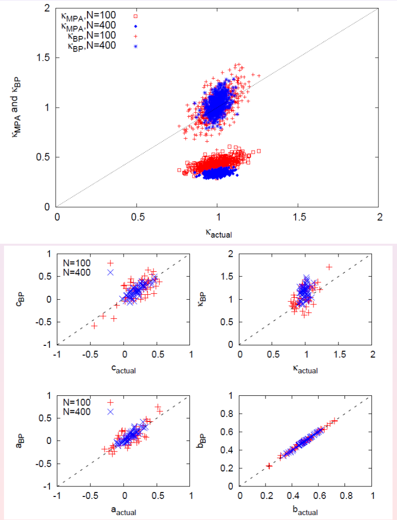
The relative error $\sqrt{\sum(\kappa_{BP/MPA}-\kappa_{\text{actual}})^2/K}/\kappa_{\text{actual}}$ in the estimates of the diffusivity over K measurements *vs.* the actual value of the diffusivity.



- κ_{actual} is the actual value of the mean-square displacement of the particles, i.e., it includes fluctuations around κ^* due to the finite number N of particles.
- The data are averaged over 1000 (for $N = 100$) and 250 (for $N = 400$) realizations and compared to relative statistical error $\sqrt{2/dN}$ (dashed horizontal lines).

Quality of BP is significantly higher!

BP vs MPA (II): Scatter Plots.



Scatter plot of the diffusivity estimated by BP and by MPA *vs.* the actual value of the diffusivity. 3D. $\kappa^* = 1$.

- The number of tracked particles is $N = 100$ (red) and $N = 400$ (blue) using 1000 measurements.
- The BP predictions correspond to the maximum of the log-likelihood.
- MPA underestimates while the cloud of BP predictions is centered around κ^*.

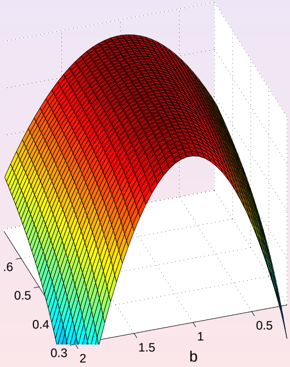
Scatter plot of the parameter estimations using the BP method in the case of a 2D incompressible flow: $a^* = 0.1$, $b^* = 0.5$, $c^* = 0.2$ and $\kappa^* = 1$.

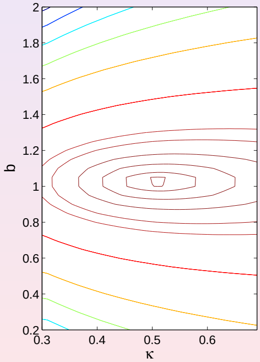
- The number of measurements is 50.

BP estimates both the parameters of the flow and the diffusivity very reliably!

Quality of the Predicition [is good]

2D. $a^* = b^* = c^* = 1$, $\kappa^* = 0.5$. $N = 200$.





- The BP Bethe free energy vs κ and b. Every point is obtained by minimizing wrt a, c
- Perfect maximum at $b = 1$ and $\kappa = 0.5$ achieved at $a_{BP} = 1.148(1)$, $b_{BP} = 1.026(1)$, $c_{BP} = 0.945(1)$, $\kappa_{BP} = 0.509(1)$.

Simplified Model to Understand BP approximation [Theory]

Random Distance Model

- Decouple N^2 distances d_i^j assuming that they are independent
- Choose a permutation π^*. $d_i^{\pi^*(i)}$ are i.i.d. Gaussian with $\kappa^* = O(1)$.
- Other distances d_i^j are drawn independently at random from a given distribution.
- Units of length are chosen so that the typical inter-particle distance is $O(1)$.

The model is solvable

- Imitates advection at $d \to \infty$ [diffusion without "geometry"]
- Similar to Random Matching model of Parisi, Mezard '85-'01
- Exact cavity [replica symmetric] analysis for distribution of messages at $N \to \infty$

- The quality of prediction improves with dimensionality increase. All asymptotical errors made by BP can be attributed to the non-vanishing inter-particle correlations.
- Observe an interesting "reconstruction" transition at $\kappa_c \approx 0.174$. At $\kappa^* < \kappa_c$ MPA is as good as BP, while at $\kappa^* > \kappa_c$ BP is a clear winner.

Outline

Summary

- The message-passing algorithms have been shown to ensure efficient, distributed and accurate learning of the parameters governing the stochastic map between two consecutive images recording the positions of many identical elements/particles.
- Introduced two techniques. MPA is based on finding the most probable trajectories of the particles between the times of the two images. BP corresponds to evaluating the probabilistically weighted sum over all possible trajectories. BP method generally gives more accurate results and its computational burden is comparable to identifying the most probable trajectories. Both methods are much more rapid than the MCMC algorithm.
- BP was shown to become exact for the simplified "random-distance" model we introduced here. In general, the effect of loops in the graphical model for the tracking problem remains nonzero even in the thermodynamic limit of a large number of tracked particles.

Path Forward

- **Implement the algorithm in an experimental setting.**
- "On-the-fly" software is the goal.
- Some generalizations are obvious (multiple sequential images, lost particles & new arrivals, other stochastic models for non-interacting particles, e.g. for other applications)
- To account for inter-particle interaction is more difficult but possible.
- Beyond BP. Loop calculus +. Preliminary analysis of the loop corrections to BP did not display any immediately visible structure, yet detailed analysis of this point is left for future work.