Learning Protein Dynamics with Metastable Switching Systems

Bharath Ramsundar\textsuperscript{1} and Vijay S. Pande\textsuperscript{2}

\textsuperscript{1)Department of Computer Science, Stanford University, Stanford CA 94305, USA}
\textsuperscript{2)Department of Chemistry, Stanford University, Stanford CA 94305, USA}

(Dated: 5 October 2016)

We introduce a machine learning approach for extracting fine-grained representations of protein evolution from molecular dynamics datasets. Metastable switching linear dynamical systems extend standard switching models with a physically-inspired stability constraint. This constraint enables the learning of nuanced representations of protein dynamics that closely match physical reality. We derive an EM algorithm for learning, where the E-step extends the forward-backward algorithm for HMMs and the M-step requires the solution of large biconvex optimization problems. We construct an approximate semidefinite program solver based on the Frank-Wolfe algorithm and use it to solve the M-step. We apply our EM algorithm to learn accurate dynamics from large simulation datasets for the opioid peptide met-enkephalin and the proto-oncogene Src-kinase. Our learned models demonstrate significant improvements in temporal coherence over HMMs and standard switching models for met-enkephalin, and sample transition paths (possibly useful in rational drug design) for Src-kinase.

I. INTRODUCTION

Understanding protein folding and conformational change is a central challenge in modern biology. Since proteins are microscopic objects, experimental techniques cannot satisfactorily resolve protein dynamics due to limited spatial and temporal resolution. Molecular dynamics (MD) simulations complement experimental understanding by providing a computational lens into the high-frequency atomic dynamics of proteins\textsuperscript{1}. MD simulations forward-integrate Newton’s equations of motion for classical approximations of the energy landscapes of biological macromolecules\textsuperscript{2}. These simulations have thousands of degrees of freedom which must be integrated over billions of timesteps to sample biologically relevant events\textsuperscript{3}. Machine learning techniques which condense MD datasets into human-comprehensible representations can provide crucial insight into protein structure and dynamics.

Several machine-learning methods have been proposed for analyzing protein trajectories. Dimensionality reduction methods such as PCA provide insight into the high-variance degrees of freedom of the system\textsuperscript{4}, but do not model the temporal structure of the data. More sophisticated methods such as Markov state models (MSMs)\textsuperscript{5} and hidden Markov models (HMMs)\textsuperscript{6} identify long-lived (metastable) conformations of the protein and model dynamics by a Markovian jump process between metastable states. These methods represent the state-of-the-art for analyzing MD trajectories and have been widely adopted\textsuperscript{7}.

Despite their strengths, HMMs and MSMs only parameterize discrete jump processes between metastable states and do not describe the complex, continuous dynamics of the underlying physical system. Consequently, these systems cannot accurately model transition paths between metastable conformations. This limitation is unfortunate, since detailed understanding of the physical transitions between metastable states could facilitate the rational design of targeted drugs\textsuperscript{8}.

Parameterizing the dynamics of a protein requires modeling the underlying physical Hamiltonian. Such modeling is complicated by the high-dimensional and intrinsically nonconvex potential energy surface of the system. HMMs and MSMS can identify the local minima of this energy landscape but make no effort to describe the interstitial terrain. Prior work has suggested using locally quadratic approximations to the Hamiltonian, which have the convenient property that the emergent dynamics are locally described by linear operators\textsuperscript{9}. The associated probabilistic model is the switching linear dynamical system\textsuperscript{10}.

Standard switching models are not well suited for modeling physical systems, since the associated linear operators must be constrained to prevent degeneracy as time goes to infinity. Earlier work has suggested constraining the spectral radius of the operators to have magnitude at most $1$\textsuperscript{11}. While this constraint is sufficient given negligible system noise, physical systems with stochastic dynamics require stronger constraints to prevent degeneracy. The proper constraint can be derived from classical notions of Lyapunov stability in control theory\textsuperscript{11}. Let $\Sigma_s$ be a fixed upper bound on the desired covariance for metastable state $s$. Let $A_s$ be a linear model for system evolution in state $s$ and let $Q_s$ model Brownian noise. The constraint $Q_s + A_s \Sigma_s A_s^T S \preceq \Sigma_s$ (that is, $\Sigma_s - Q_s - A_s \Sigma_s A_s^T S$ is a positive semidefinite matrix) guarantees that the learned model for metastable state $s$ will have covariance upper bounded by $\Sigma_s$.

We define a metastable switching linear dynamical system to be a switching model with covariance constraint $Q_s + A_s \Sigma_s A_s^T \preceq \Sigma_s$ and spectral norm constraint $\|A_s\|_2 < 1$ for each metastable state $s$. Section II introduces the graphical model associated with a metastable switching system and proves the sufficiency of our stability constraints. Section III establishes a learn-
The linear operator $A_s$ and affine shift vector $b_s$ control the evolution of the system in metastable state $s_t$. The terms $w_{s_t}$ are noise parameters drawn from distribution $\mathcal{N}(0, Q_s)$, where $Q_s$ is a “local” covariance matrix that corresponds to Brownian noise. To guarantee that the model is stable, iterating the transformation $A_s x_{t-1} + b_s x_t + w_{s_t}$ must not lead to unbounded mean or covariance.

Let the initial state be $x_0$ and the metastable state $s$. The distribution of $X_1$ is $\mathcal{N}(A_s x_0 + b_s Q_s)$, while the distribution of $X_2$ is $\mathcal{N}(A_s^2 x_0 + A_s b_s + b_s Q_s + A Q_s A^T)$. Iterating, the distribution of $X_n$ is

$$
\mathcal{N}
\left(A_s^n x_0 + \sum_{k=0}^{n-1} A_s^k b_s, \sum_{k=0}^{n-1} A_s^k Q_s (A_s^k)^T\right).
$$

To achieve an accurate model, this distribution must match the true distribution of metastable state $s$. Let $\mathcal{N}(\mu_s, \Sigma_s)$ be a Gaussian model of the distribution of physical conformations associated with metastable state $s$. If $b_s = \mu_s - A_s \mu_s$, the mean of the iterated distribution above converges to $\mu_s$ if the spectral norm of $A_s$ is less than 1.

**Lemma 1.** If $\|A_s\|_2 < 1$, and $b_s = \mu_s - A_s \mu_s$, then $A_s^n x_0 + \sum_{k=0}^{n-1} A_s^k b_s$ converges to $\mu_s$ as $n \to \infty$.

**Proof.** Since $\|A_s\|_2 < 1$, the summation $\sum_{k=0}^{\infty} A_s^k$ converges to $(I - A_s)^{-1}$, and the limit $\lim_{n \to \infty} A_s^n$ equals 0. Thus, $\lim_{n \to \infty} (A_s^n x_0 + \sum_{k=0}^{\infty} A_s^k b_s)$ equals $(I - A_s)^{-1} b_s$, which in turn equals $\mu_s$ if and only if $b_s = \mu_s - A_s \mu_s$. □

The next lemma proves that enforcing constraint $Q_s + A_s \Sigma_s A_s^T \preceq \Sigma_s$ guarantees that the iterated distribution has covariance upper bounded by $\Sigma_s$ (recall that the constraint $A \preceq B$ denotes that $B - A$ is a positive semidefinite matrix).

**Lemma 2.** Assume that $Q_s + A_s \Sigma_s A_s^T \preceq \Sigma_s$ and that $Q_s \preceq \Sigma_s$. Then for all $n \in \mathbb{N}$, $\sum_{k=0}^{n-1} A_s^k Q_s (A_s^k)^T \preceq \Sigma_s$.

**Proof.** Since $Q_s \preceq \Sigma_s$, it follows that $Q_s + A_s \Sigma_s A_s^T \preceq Q_s + A_s \Sigma_s A_s^T \preceq \Sigma_s$. The result follows by induction. □

### III. LEARNING

A metastable switching model is parameterized by choice of linear operators $\{A_s\}$, affine shifts $\{b_s\}$, local covariances $\{Q_s\}$, and transition matrix $T$. The complete-data likelihood for the model is

$$
\mathcal{L}(\{S_t\}, \{X_t\}| T, \{A_s\}, \{b_s\}, \{Q_s\}) = 
\prod_{t=1}^{T} \prod_{s_{t-1}, s_t} \prod_{i=1}^{n} \mathcal{N}(x_t | A_{s_{t-1}} x_{t-1} + b_{s_{t-1}}, Q_{s_{t-1}})
$$

We use expectation-maximization to learn the model parameters by optimizing the log-likelihood of observed data $\{X_t\}$. Section IIIA describes the inference procedure for the E-step, and section IIIB reduces the M-step to the solution of a biconvex optimization problem.

---

**FIG. 1:** Graphical model for metastable switching linear dynamical systems

**II. METASTABLE SWITCHING LINEAR DYNAMICAL SYSTEMS**

Let $\{X_t\}$ be an observed time series in $\mathbb{R}^D$, and let $\{S_t\}$ be a hidden time series of latent states in $\{1, \ldots, K\}$. For molecular dynamics trajectory data, each hidden state corresponds to a metastable state of the physical system, while each observed state corresponds to a physical conformation of the protein. A metastable switching linear dynamical system is a generative probabilistic model over the $S_t$ and $X_t$. The graphical model representation of this model is shown in Figure 1.

The hidden states $S_t$ evolve in a Markovian manner according to probability matrix $T \in \mathbb{R}^{K \times K}$, where $T_{s_{t-1}, s_{t}}$ is the probability of hidden state $s_{t-1}$ transitioning to hidden state $s_{t}$. The observed states $X_t$ evolve by an affine transformation of $X_{t-1}$ parameterized by hidden state $s_t$ (we follow that convention that upper case $X_t$ denotes the random variable and lower case $x_t$ a particular value):

$$
x_t = A_{s_{t}} x_{t-1} + b_{s_{t}} + w_{s_{t}},
$$

where $w_{s_{t}}$ are noise parameters drawn from distribution $\mathcal{N}(0, Q_{s_{t}})$, which is a “local” covariance matrix that corresponds to Brownian noise. To guarantee that the model is stable, iterating the transformation $A_{s_{t}} x_{t-1} + b_{s_{t}} + w_{s_{t}}$ must not lead to unbounded mean or covariance.

Let the initial state be $x_0$ and the metastable state $s$. The distribution of $X_1$ is $\mathcal{N}(A_{s_{0}} x_0 + b_{s_{0}} Q_{s_{0}})$, while the distribution of $X_2$ is $\mathcal{N}(A_{s_{0}}^2 x_0 + A_{s_{0}} b_{s_{0}} + b_{s_{0}} Q_{s_{0}} + A Q_{s_{0}} A^T)$. Iterating, the distribution of $X_n$ is

$$
\mathcal{N}
\left(A_{s_{0}}^n x_0 + \sum_{k=0}^{n-1} A_{s_{0}}^k b_{s_{0}}, \sum_{k=0}^{n-1} A_{s_{0}}^k Q_{s_{0}} (A_{s_{0}}^k)^T\right).
$$

To achieve an accurate model, this distribution must match the true distribution of metastable state $s$. Let $\mathcal{N}(\mu_s, \Sigma_s)$ be a Gaussian model of the distribution of physical conformations associated with metastable state $s$. If $b_s = \mu_s - A_s \mu_s$, the mean of the iterated distribution above converges to $\mu_s$ if the spectral norm of $A_s$ is less than 1.

**Lemma 1.** If $\|A_s\|_2 < 1$, and $b_s = \mu_s - A_s \mu_s$, then $A_s^n x_0 + \sum_{k=0}^{n-1} A_s^k b_s$ converges to $\mu_s$ as $n \to \infty$.

**Proof.** Since $\|A_s\|_2 < 1$, the summation $\sum_{k=0}^{\infty} A_s^k$ converges to $(I - A_s)^{-1}$, and the limit $\lim_{n \to \infty} A_s^n$ equals 0. Thus, $\lim_{n \to \infty} (A_s^n x_0 + \sum_{k=0}^{\infty} A_s^k b_s)$ equals $(I - A_s)^{-1} b_s$, which in turn equals $\mu_s$ if and only if $b_s = \mu_s - A_s \mu_s$. □

The next lemma proves that enforcing constraint $Q_s + A_s \Sigma_s A_s^T \preceq \Sigma_s$ guarantees that the iterated distribution has covariance upper bounded by $\Sigma_s$ (recall that the constraint $A \preceq B$ denotes that $B - A$ is a positive semidefinite matrix).

**Lemma 2.** Assume that $Q_s + A_s \Sigma_s A_s^T \preceq \Sigma_s$ and that $Q_s \preceq \Sigma_s$. Then for all $n \in \mathbb{N}$, $\sum_{k=0}^{n-1} A_s^k Q_s (A_s^k)^T \preceq \Sigma_s$.

**Proof.** Since $Q_s \preceq \Sigma_s$, it follows that $Q_s + A_s \Sigma_s A_s^T \preceq Q_s + A_s \Sigma_s A_s^T \preceq \Sigma_s$. The result follows by induction. □

### III. LEARNING

A metastable switching model is parameterized by choice of linear operators $\{A_s\}$, affine shifts $\{b_s\}$, local covariances $\{Q_s\}$, and transition matrix $T$. The complete-data likelihood for the model is

$$
\mathcal{L}(\{S_t\}, \{X_t\}| T, \{A_s\}, \{b_s\}, \{Q_s\}) = 
\prod_{t=1}^{T} \prod_{s_{t-1}, s_t} \prod_{i=1}^{n} \mathcal{N}(x_t | A_{s_{t-1}} x_{t-1} + b_{s_{t-1}}, Q_{s_{t-1}})
$$

We use expectation-maximization to learn the model parameters by optimizing the log-likelihood of observed data $\{X_t\}$. Section IIIA describes the inference procedure for the E-step, and section IIIB reduces the M-step to the solution of a biconvex optimization problem.
A. E-step

The E-step for a metastable system estimates hidden states \( \{S_t\} \) given model parameters \((T, \{A_s\}, \{b_s\}, \{Q_s\})\) and observed data \( \{X_t\} \) by using a variant of the forward-backward algorithm for Gaussian HMMs. We derive the recursive forward updates that compute \( \alpha(s_t) = P(s_t | x_{1:t}) \), the posterior on hidden state \( S_t \) given observed data \( x_{1:t} \).

\[
\alpha(s_{t+1}) = \mathbb{P}(s_{t+1} | x_{1:t+1}) \\
\propto \mathbb{P}(x_{t+1} | s_{t+1}, x_t) \sum_{s_t} \mathbb{P}(s_{t+1} | s_t) \mathbb{P}(s_t | x_{1:t}) \\
= \mathcal{N}(x_{t+1} | A_{s_{t+1}} x_t + b_{s_{t+1}}, Q_{s_{t+1}}) \sum_{s_t} T_{s_t, s_{t+1}} \alpha(s_t)
\]

We analogously derive the backward updates for \( \beta(s_t) = \mathbb{P}(x_{t+1:T} | s_t, x_t) \), the probability of the future data conditioned on \( s_t \) and \( x_t \).

\[
\beta(s_t) = \mathbb{P}(x_{t+1:T} | s_t, x_t) \\
= \sum_{s_{t+1}} \mathbb{P}(x_{t+1:T} | s_{t+1}, x_t) \mathbb{P}(x_{t+1} | s_{t+1}, x_t) \\
\times \mathbb{P}(s_{t+1} | s_t, x_t) \\
= \sum_{s_{t+1}} \beta(s_{t+1}) \mathcal{N}(x_{t+1} | A_{s_{t+1}} x_t + b_{s_{t+1}}, Q_{s_{t+1}}) T_{s_t, s_{t+1}}.
\]

In both derivations, we use the fact that \( d \)-separation implies that

\[
\mathbb{P}(s_{t+1} | s_t, x_t) = \mathbb{P}(s_{t+1} | s_t) = T_{s_t, s_{t+1}}.
\]

We compute the posterior probability on hidden states \( \gamma_{s_t}(t) \) as for a HMM:

\[
\gamma_{s_t}(t) = \mathbb{P}(s_t | x_{1:T}) \propto \beta(s_t) \alpha(s_t)
\]

Let \( x_{t+1}' = A_{s_t} x_t + b_{s_t} \). Then the joint transition probability \( \xi_{s_t, s_{t+1}} \) is

\[
\xi_{s_t, s_{t+1}} = \mathbb{P}(x_{t+1}' | s_{t+1}) \\
= \frac{\alpha(s_t) T_{s_t, s_{t+1}} \beta(s_{t+1}) \mathcal{N}(x_{t+1} | x_{t+1}', Q_{s_{t+1}})}{\sum_s \alpha(s) \beta(s)}
\]

1. Implementation

The E-step is implemented as a multithreaded CPU application. The quantities \( \alpha, \beta \) are computed in parallel across MD trajectories during the E-step using OpenMP. The largest portion of the run time is spent in log-sum-exp operations, which are manually vectorized with SSE2 intrinsics for SIMD architectures.

B. M-step

The M-step selects model parameters \((T, \{A_s\}, \{Q_s\}, \{b_s\})\) that optimize the complete-data log-likelihood for the model. We assume that for each metastable state \( s \), we are given a Gaussian \( \mathcal{N}(\mu_s, \Sigma_s) \) fitted to the data from this state. The mean \( \mu_s \) and \( \Sigma_s \) are used to constrain the optimization problem to ensure stability, but these quantities are not themselves estimated in the M-step; they should be estimated in a preliminary step by fitting a Gaussian mixture model to the data.

1. Learning \( T \)

The transition matrix \( T \) is learned as for the hidden Markov model:

\[
T_{i,j} = \frac{\sum_t \xi_{i,t}(t)}{\sum_t \gamma_i(t)}
\]

2. Objective for a fixed metastable state \( s \)

In principle, the terms \( \{A_s\}, \{Q_s\}, \{b_s\} \) should be chosen to maximize the log-likelihood of the observed data. Let \( x_{t+1}' = A_s x_t + b_s \).

\[
\log \mathcal{L}(\{X_t\} | T, \{A_s\}, \{b_s\}, \{Q_s\}) = \\
\int \sum_{t=1}^{T} \left[ T_{s_{t-1}, s_t} + \sum_{t=1}^{T} \log \mathcal{N}(x_t | x_{t+1}', Q_{s_t}) \right] dS_{1:T}.
\]

The objective above is intractable, so we instead exploit an approximation of the objective for \((A_s, Q_s, b_s)\) that weights the terms for time \( t \) with posterior likelihood \( \gamma_s(t) \). With the stability constraints added, the joint optimization problem becomes

\[
\text{minimize } \quad \log \det(Q_s) \left( \sum_t \gamma_s(t) \right) \\
+ \sum_t \gamma_s(t) (x_t - A_s x_{t-1} - b_s)^T Q_s^{-1} (x_t - A_s x_{t-1} - b_s) \\
\text{subject to } \quad Q_s + A_s \Sigma_s A_s^T \preceq \Sigma_s, \quad \|A_s\|_2 < 1 \\
A_s \mu_s + b_s = \mu_s.
\]

Note that constraint \( A_s \mu_s + b_s = \mu_s \) uniquely determines \( b_s \). Thus, the optimization above depends only on \( A_s \) and \( Q_s \). We show in the following sections that this objective is convex in both variables.

3. Learning \( A_s \)

Consider the objective with \( Q_s \) and \( b_s \) fixed. The log det can be dropped, and the cyclic invariance of the trace can be used to factor out \( Q_s^{-1} \):

\[
\text{Tr} \left( Q_s^{-1} \sum_t \gamma_s(t) (x_t - A_s x_{t-1} - b_s)(x_t - A_s x_{t-1} - b_s)^T \right).
\]
The summation can be simplified by dropping terms not
dependent on \( A_s \):

\[
\text{Tr} \left( Q_s^{-1} \left[ -B_s A_s^T - A_s B_s^T + A_s E_s A_s^T + A_s C_s^T + C_s A_s^T \right] \right).
\]

In this expression, we use sufficient statistics matrices
calculated from the data in the E-step:

\[
C_s = b \sum_t \gamma_s(t)x_t^T  \\
B_s = \sum_t \gamma_s(t)x_t x_t^T  \\
E_s = \sum_t \gamma_s(t)x_t x_t^T - 1
\]

Consider the constraint \( Q_s + A_s \Sigma_s A_s^T \leq \Sigma_s \). This expression can be rewritten by using the Schur complement; since the covariance upper bound \( \Sigma_s \) is positive definite, the matrix

\[
\begin{pmatrix}
\Sigma_s - Q_s & A_s \\
A_s^T & \Sigma_s \\
\end{pmatrix}
\]

is positive semidefinite if and only if \( Q_s + A_s \Sigma_s A_s^T \leq \Sigma_s \). Similarly, the constraint \( \|A_s\|_2 < 1 \) can be rewritten as a Schur complement; the following matrix is positive semidefinite if and only if \( \|A_s\|_2 \leq \eta \), where \( \eta < 1 \) is a fixed constant:

\[
\begin{pmatrix}
\eta I & A_s \\
A_s^T & I \\
\end{pmatrix}
\geq 0.
\]

Now, the optimization problem for \( A_s \) is a quadratic matrix
program with two linear matrix inequalities. Let \( F_s = C_s - B_s \). Then

\[
\text{minimize}_{A_s} \quad \text{Tr} \left( Q_s^{-1} \left( F_s A_s^T + A_s F_s^T + A_s E_s A_s^T \right) \right) \\
\text{subject to} \quad \begin{pmatrix}
\Sigma_s - Q_s & A_s \\
A_s^T & \Sigma_s^{-1} \\
\end{pmatrix} \geq 0, \quad \begin{pmatrix}
\eta & A_s \\
A_s^T & I \\
\end{pmatrix} \geq 0.
\]

4. Learning \( Q_s \)

Consider the objective with \( A_s \) and \( b_s \) fixed. Applying the cyclic invariance of the trace simplifies the objective to

\[
\log \det(Q_s)g_s + \text{Tr} \left( Q_s^{-1} F_s \right),
\]

where \( F_s \) and \( g_s \) are sufficient statistics matrices computed from the data:

\[
F_s = \sum_t \gamma_s(t)(x_t - A_s x_{t-1} - b_s)(x_t - A_s x_{t-1} - b_s)^T \\
g_s = \left( \sum_t \gamma_s(t) \right).
\]

We simplify the objective further by applying the change of variable \( R_s = Q_s^{-1} \):

\[
- \log \det R_s g_s + \text{Tr}(R_s F_s).
\]

The stability constraint becomes \( R_s^{-1} + A_s \Sigma_s A_s^T \leq \Sigma_s \). Applying the Schur complement as before, the optimization problem for \( Q \) becomes the following semidefinite program:

\[
\text{minimize}_{R_s} \quad - \log \det(R_s)g_s + \text{Tr}(R_s F_s) \\
\text{subject to} \quad \begin{pmatrix}
\Sigma_s - A_s \Sigma_s A_s^T & I_n \\
I_n & R_s \\
\end{pmatrix} \geq 0.
\]

IV. SOLVING FOR \( \{A_s\}, \{Q_s\} \) WITH A FRANK-WOLFE ALGORITHM

The learning problems for \( A_s \) and \( Q_s \) are convex pro-
grams with semidefinite cone constraints. Although interior point solvers routinely solve small cone programs\(^\text{12}\), these methods require the construction of large Hessian and inverse Hessian matrices. Thus, we use a first-order Frank-Wolfe algorithm\(^\text{17}\) to solve for \( A_s \) and \( Q_s \) in large protein systems. We extend prior work that uses a Frank-Wolfe method to solve semidefinite programs (SDPs)\(^\text{13}\).

A. Generalized penalty

The Frank-Wolfe SDP solver encodes linear matrix in-
equality constraints within a penalty function. The min-
ima of the penalty function lies below some threshold if and only if the SDP is approximately feasible. A bi-
ary search then enables the solution of general SDPs with logarithmic overhead. This algorithm cannot solve for \( A_s \) and \( Q_s \), since the objective functions are nonlin-
ear. We introduce a new penalty function which resolves this issue. Let \( f_1, \ldots, f_N \) be convex functions on \( \mathbb{R}^N \) with associated inequality constraints \( f_i(X) \leq 0 \), and let \( g_j(X) \) be affine functions on \( \mathbb{R}^N \) with associated equality constraints \( g_j(X) = 0 \). Define a log-sum-exp penalty function

\[
\Phi(\{f_i\}, \{g_j\})(X) = \frac{1}{M} \log \left( \sum_{i=1}^n \exp(M f_i(X)) + \sum_{j=1}^m \exp \left( M g_j(X)^2 \right) \right)
\]

Note that \( \Phi(\{f_i\}, \{g_j\}) \) (or \( \Phi \) for short) is a convex function by the rules of convex composition\(^\text{18}\), since the \( f_i \) and \( g_j^2 \) are convex and the log-sum-exp function is con-
vex and nondecreasing in each argument. Penalty \( \Phi \) lies below some threshold \( \epsilon \) if and only if \( X \) approximately satisfies constraints \( f_i(X) \leq 0 \) and \( g_j(X) = 0 \).

Lemma 3. Let \( M = \log(n + m)/\epsilon \). For all \( X \in \mathbb{R}^{N \times N} \), if objective \( \Phi(X) \) is upper bounded by \( \epsilon \), then \( X \) satisfies
constraints \( f_i(X) \leq \epsilon \) and \( g_j(X)^2 \leq \epsilon \). Conversely, if \( X \) is such that \( f_i(X) \leq \epsilon \) and \( g_j(X)^2 \leq \epsilon \) for all \( i \) and \( j \), then \( \Phi(X) \leq 2\epsilon \).

**Proof.** Define \( \phi(x) = \frac{1}{n} \log \sum_{i=1}^n \exp(Mx_i) \). For \( M \geq 0 \), the function \( \phi \) is a smooth approximation to the max function\(^{13}\):

\[
\max_i x_i \leq \phi(x) \leq \max_i x_i + \frac{\log \ell}{M}
\]

Thus if \( \Phi(X) \leq \epsilon \), then \( X \) satisfies condition \( \max_{i,j} \{ f_i(X), g_j(X)^2 \} \leq \Phi(X) \leq \epsilon \). Conversely, suppose \( X \) is such that \( f_i(X) \leq \epsilon \) and \( g_j^2(X) \leq \epsilon \). The upper bound above, combined with choice of \( M \), guarantees that \( \Phi(X) \leq 2\epsilon \).

\[\square\]

### B. Feasibility Search

#### Algorithm 1 Feasibility search

```
1: procedure Feasibility-Search(h, \{f_i\}, \{g_j\}, \epsilon, R, K)
2: \triangleright h objective, \( f_i(X) \leq 0, g_j(X) = 0 \), constraints, \( \epsilon > 0 \)
3: error threshold
4: \( X \leftarrow \) Frank-Wolfe(\( \Phi(\{f_i\}, \{g_j\}), R, K \))
5: \textbf{if} \( \Phi(\{f_i\}, \{g_j\})(X) < \epsilon \) \textbf{then}
6: \( U \leftarrow h(X), \eta \leftarrow 1 \triangleright \eta \) is step size
7: \textbf{while} \( \eta \geq \epsilon \) \textbf{do}
8: \( h_{U-\eta} \leftarrow \lambda Y : h(Y) - (U - \eta) \)
9: \( X_{U-\eta} \leftarrow \) Frank-Wolfe(\( \Phi(\{h_{U-\eta}, f_i\}, \{g_j\}), R, K \))
10: \textbf{if} \( \Phi(\{h_{U-\eta}, f_i\}, \{g_j\})(X_{U-\eta}) < \epsilon \) \textbf{then}
11: \( X \leftarrow X_{U-\eta}, U \leftarrow h(X), \eta \leftarrow 2\eta \)
12: \textbf{else}
13: \( \eta \leftarrow \frac{1}{2}\eta \)
14: \textbf{return} Success, \( X \)
15: \textbf{return} Fail
```

#### Algorithm 2 Frank-Wolfe

```
procedure Frank-Wolfe(\( \Phi, R, K \))
\triangleright R upper bound on \( \text{tr}(X) \), \( K \) number of desired descent steps
1: \( X \leftarrow \) RESCALE(\( X \))
2: for \( k \leftarrow 1 \ldots K \) \textbf{do}
3: \( v_k \leftarrow \) APPROX-EV(\( -\nabla \Phi(X) \))
4: Line search for \( \gamma \)
5: \( X \leftarrow (1 - \gamma)X + \gamma v_k v_k^T \)
6: \textbf{return} \( X \)
```

We introduce a feasibility search method in algorithm 1 which optimizes a convex objective \( h \) while attempting to satisfy constraints \( f_i(X) \leq 0 \) and \( g_j(X) = 0 \). The procedure attempts to find any feasible \( X \). If it succeeds, then \( U = h(X) \) is an upper bound for the minima. The method attempts to reduce \( U \) by transforming \( h \) into constraint \( h_{U-\eta}(Y) = h(Y) - (U - \eta) \leq 0 \), where \( \eta \) is a step-size. A feasibility problem is solved with added constraint \( h_{U-\eta}(X) \leq 0 \). Successes update \( U \) and \( X \) and double the size of \( \eta \), while failures halve the size of \( \eta \). The search halts when \( \eta \) falls below threshold \( \epsilon \). Algorithm 1 can efficiently solve for \( A_s \) and \( Q_s \) given hundreds of features.

Algorithm 2, restated here for completeness, minimizes convex function \( \Phi(X) \) under the constraints that \( \text{tr}(X) \leq R \) over the course of \( K \) descent steps\(^{13}\). The problems for \( A_s \) and \( Q_s \) each yield natural estimates for \( R \). The primitive RESCALE applies a change of variable to transform the constraint \( \text{tr}(X) \leq R \) into the constraint \( \text{tr}(X) = 1 \) by rescaling and adding a slack variable. The primitive APPROX-EV computes the eigenvector associated with the largest eigenvalue of its argument. The Lanczos algorithm can perform this operation in time \( O(d^2) \) per iteration\(^{13}\). The number of iterations required is \( O(\log(N)/\sqrt{C/\epsilon}) \), where \( C \) is any upper bound on the spectral norm of \( X \).

1. **Numerical Issues**

If the gradient at \( X \) has largest eigenvalue near zero, the Lanczos algorithm can fail. In these cases, we shift the matrix upwards by a multiple of the identity\(^{19}\). If the Lanczos algorithm fails to converge even after shifting, we fall back to a slower but more stable LAPACK implementation of a divide-and-conquer eigenvalue algorithm.

## V. EXPERIMENTS

In this section, we learn metastable switching models from two MD simulation datasets. Our results indicate that metastable switching models can learn realistic models of protein dynamics.

### A. Met-enkephalin

Met-enkephalin is a small, naturally-occurring opioid peptide with five amino-acids and 75 atoms\(^{14}\). To study the dynamics of met-enkephalin, we used a collection of publicly available MD trajectories\(^{20}\). We superposed the
FIG. 3: Comparison of trajectories. Blue to red and small to large denote increasing time.

raw trajectories upon a starting structure to eliminate rotational tumbling. We encoded the \((x, y, z)\) coordinates of all atoms in the superposed trajectories to obtain a time-series with \(10^6\) data points, each with \(d = 225\) features.

FIG. 4: Cartoon of Src-kinase

Figure 2 displays the structure of met-enkephalin, and Figure 3 compares trajectories from the original MD dataset (far-left) to trajectories sampled from a learned 2-state metastable switching linear dynamical system (MSLDS, center left), trajectories sampled from a learned 2-state HMM (center right), and trajectories sampled from a learned 2-state switching linear dynamical system (SLDS, right). The plots are projected onto chemically relevant order parameters for visualization. The scattered points shift in color from blue to red (and from small to large) as time increases. The MSLDS preserves the temporal coherence of the original data, while the HMM does not. The SLDS degenerates and does not capture the true distribution of the data.

B. Src-Kinase

Protein kinases are enzymes critical to the cellular regulation network. The Src-Kinase is a member of this family with 262 amino acids. We obtained a dataset that samples 550\(\mu\)s of dynamics from this system for a total of 108GB of data\(^6\). To reduce the size of the feature space (the original system has over \(4 \times 10^5\) atoms including waters), we selected the \((x, y, z)\) coordinates of 121 critical atoms for a total of \(d = 363\) features. We then superposed upon a starting structure and subsampled every tenth data point from the featurized dataset, which reduced the data size to 1.8GB.

Figure 4 displays the Src-kinase enzyme. Figure 5 displays multiple sampled trajectories from a learned 3-state MSLDS model. The colors rise from blue to red as time increases. The model identifies three crucial regions previously discovered in the energy landscape\(^15\). More interestingly, the MSLDS model samples transition paths between these metastable regions, a feat not achievable with HMMs or MSMs. Such paths may be useful for the rational design of kinase inhibitors\(^15\).

VI. DISCUSSION AND CONCLUSION

This work presents an approach for learning protein dynamics from data and provides three major contributions: (1) The introduction of the metastable switching model, which contains physically inspired constraint \(Q_s + A_s \Sigma_s A_s^T \preceq \Sigma_s\) (SLDS, right), and the derivation of associated inference and learning algorithms; (2) The extension of Frank-Wolfe algorithms to a broader class of convex programs; (3) The demonstration that metastable switching models can accurately learn the temporal dynamics of met-enkephalin and generate transition paths between metastable states of Src-kinase. Future work might apply metastable switching models to other physical datasets, such as temporal data from quantum-mechanical simulations.

---

\(^1\) Ken A Dill, Sarina Bromberg, Kaizhi Yue, Hue Sun Chan, Klaus M Fiebig, David P Yee, and Paul D Thomas. Principles of protein folding – a perspective from simple exact models. *Protein Science*, 4(4):561–602, 1995.

\(^2\) Kyle A Beauchamp, Yu-Shan Lin, Riju Das, and Vijay S Pande. Are protein force fields getting better? a systematic benchmark on 524 diverse NMR measurements. *Journal of chemical theory and computation*, 8(4):1409–1414, 2012.
3. David E Shaw et al. Anton, a special-purpose machine for molecular dynamics simulation. Communications of the ACM, 51(7):91–97, 2008.

4. Alexandros Altis, Phuong H Nguyen, Rainer Hegger, and Gerhard Stock. Dihedral angle principal component analysis of molecular dynamics simulations. The Journal of chemical physics, 126(24):244111, 2007.

5. Gregory R Bowman, Xuhui Huang, and Vijay S Pande. Using generalized ensemble simulations and Markov state models to identify conformational states. Methods, 49(2):197–201, 2009.

6. Robert T McGibbon, Bharath Ramsundar, Mohammad M Sultan, Gert Kiss, and Vijay S Pande. Understanding protein dynamics with L1-regularized reversible hidden Markov models. arXiv preprint arXiv:1405.1444, 2014.

7. Kai J Kohlhoff, Diwakar Shukla, Morgan Lawrenz, Gregory R Bowman, David E Konerding, Dan Belov, Russ B Altman, and Vijay S Pande. Cloud-based simulations on Google Exacycle reveal ligand modulation of GPCR activation pathways. Nature chemistry, 6(1):15–21, 2014.

8. Michael L West and David P Fairlie. Targeting HIV-1 protease: a test of drug-design methodologies. Trends in pharmacological sciences, 16(2):67–75, 1995.

9. Illia Horenko and Christof Schütte. Likelihood-based estimation of multidimensional Langevin models and its application to biomolecular dynamics. Multiscale Modeling & Simulation, 7(2):731–773, 2008.

10. Kevin P Murphy. Switching kalman filters. Technical report, Citeseer, 1998.

11. Sajid M Siddiqi, Byron Boots, and Geoffrey J Gordon. A constraint generation approach to learning stable linear dynamical systems. Technical report, DTIC Document, 2008.

12. Christoph Helmberg, Franz Rendl, Robert J Vanderbei, and Henry Wolkowicz. An interior-point method for semidefinite programming. SIAM Journal on Optimization, 6(2):342–361, 1996.

13. Elad Hazan. Sparse approximate solutions to semidefinite programs. In LATIN 2008: Theoretical Informatics, pages 306–316. Springer, 2008.

14. Michael Com, Peter H Seeburg, John Adelman, Lee Eiden, and Edward Herbert. Primary structure of the human met- and leu-enkephalin precursor and its mRNA. Nature, 295(5851):663–666, 1982.

15. Diwakar Shukla, Yilin Meng, Benoît Roux, and Vijay S. Pande. Activation pathway of Src kinase reveals intermediate states as novel targets for drug design. Nature Communications, in press.

16. Lawrence Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2):257–286, 1989.

17. Kenneth L Clarkson. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. ACM Transactions on Algorithms (TALG), 6(4):63, 2010.

18. Stephen P Boyd and Lieven Vandenberghe. Convex optimization. Cambridge university press, 2004.

19. Martin Jaggi, Marek Sulovsk, et al. A simple algorithm for nuclear norm regularized problems. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), pages 471–478, 2010.

20. Met-enkephalin MD Trajectories. http://figshare.com/articles/Met_enkephalin_MD_Trajectories/1026324. Accessed: 2014-06-01.