Fast Direct Policy Evaluation using Multiscale Analysis of Markov Diffusion Processes

Mauro Maggioni MAURO.MAGGIONI@YALE.EDU
Department of Mathematics, Yale University, P.O. Box 208283, New Haven, CT, 06511, U.S.A.

Sridhar Mahadevan MAHADEVA@CS.UMASS.EDU
Department of Computer Science, University of Massachusetts at Amherst, 140 Governors Drive, MA, 01003, U.S.A.

Abstract

Policy evaluation is a critical step in the approximate solution of large Markov decision processes (MDPs), typically requiring $O(|S|^3)$ to directly solve the Bellman system of $|S|$ linear equations (where $|S|$ is the state space size in the discrete case, and the sample size in the continuous case). In this paper we apply a recently introduced multiscale framework for analysis on graphs to design a faster algorithm for policy evaluation. For a fixed policy π, this framework efficiently constructs a multiscale decomposition of the random walk P^π associated with the policy π. This enables efficiently computing medium and long term state distributions, approximation of value functions, and the *direct* computation of the potential operator $(I-\gamma P^\pi)^{-1}$ needed to solve Bellman's equation. We show that even a preliminary non-optimized version of the solver competes with highly optimized iterative techniques, requiring in many cases a complexity of $O(|S|)$.

1. Introduction

In this paper we apply a novel framework for multiscale analysis of Markov diffusion processes on graphs to efficiently solving certain classes of Markov Decision Processes (MDPs). The approach is based on learning a multiscale tree of *wavelet*-type basis functions on the state space of a MDP, which allows efficient hierarchical representation of value functions, and yields a fast algorithm for the direct solution of the Bellman equation for policy evaluation. The paper focuses on policy evaluation primarily, and the resulting method can be easily incorporated in approximate or exact policy iteration (PI) algorithms, such as those in (Lagoudakis & Parr, 2003; Mahadevan, 2005; Mahadevan & Maggioni, 2005).

Bellman's equation (discussed in the next section) usually involves the solution of a (typically sparse) linear system

$$V^\pi = R + \gamma P^\pi V^\pi$$

of size $|S|$, where S is the state space (or a set of sample states, in the continuous case). Here V^π is the unknown value function, R and P^π are given (and may change over each step of policy iteration). We can divide the approaches for solving this equation in two families: direct or iterative.

(a) *Direct solution* involves the computation of the inverse $(I-\gamma P^\pi)^{-1}$. While the matrix $I-\gamma P$ is usually sparse, its inverse is in general a full matrix, and its computation usually takes time $\mathcal{O}(|S|^3)$. This is in general infeasible on large problems. However it presents certain computational advantages: very stable algorithms exist, computations can be done to very high precision, and once the inverse matrix has been computed, it can be applied in time $\mathcal{O}(|S|^2)$ to any reward vector R, or time $\mathcal{O}(|S|)$ if R is sparse (a constant number of nonzero entries, small compared to $|S|$).

(a) *Iterative solution* involves iterative techniques, such as conjugate gradient or value iteration, which compute $(I-\gamma P^\pi)^{-1}R$ for a given R. These techniques have worst case complexity $\mathcal{O}(|S|^2)$ for sparse transition matrices, $\mathcal{O}(|S|)$ when the problem is well-conditioned and only low-precision is required. No structure of the problem or of previous solutions is constructed, so the computation

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

has to be repeated for different reward vectors.

We propose to use a fundamentally different approach, based on the multiscale analysis on graphs introduced in (Coifman & Maggioni, 2004), that yields a *direct* solution in time $\mathcal{O}(|S|)$, for certain classes of transition matrices P^π. This is surprising because, as we observed before, just writing down all the entries of the full inverse matrix would take time $\mathcal{O}(|S|^2)$. However, observe that the goal is not to compute the entries of this matrix, but to compute a structure that allows rapidly computing $(I-\gamma P^\pi)^{-1}R$ for any given R. The proposed approach constructs a (multiscale) structure that performs this computation in only $\mathcal{O}(|S|)$ operations. Related ideas are at the core of the Fast Fourier Transform, which is a full matrix multiplication by a vector, and hence would seem to necessarily require $\mathcal{O}(|S|^2)$ operations, but it can actually be performed in $\mathcal{O}(|S|)$ operations by factoring the full matrix in a product of matrices. A related principle is also behind the Fast Multipole Method (Greengard & Rokhlin, 1987), which evaluates the product of full matrices arising from potential theory by vectors in time $\mathcal{O}(|S|)$.

The algorithm consists of two parts:

(i) a *pre-computation* step, that depends on the structure of the state space and on the policy. The result of this step is a multiscale hierarchical decomposition of the value function space over the state space, and a multiscale compression of powers of the transition matrix over the state space. This computation, for classes of problems of interest in applications, has complexity $\mathcal{O}(|S|)$, and complexity $\mathcal{O}(|S|^3)$ in general.

(ii) an *inversion* step which uses the multiscale structure built in the pre-computation step to efficiently compute the solution of Bellman's equations for a given reward function. This phase of the computation has complexity $\mathcal{O}(|S|)$ for many problems of practical importance where the transition matrix is *diffusion*-like (defined precisely below). The constants in front of this asymptotic complexity are much smaller than those in the pre-computation step.

In (Mahadevan & Maggioni, 2005) it was shown that the basis functions constructed in the pre-computation step are also very useful in approximating the value function and can outperform parametric bases like polynomials and RBF's in policy improvement.

The class of problems for which the complexity of the method is linear (up to logarithmic factors) includes state spaces that can be represented by a finite directed weighted graph, with all the vertices of "small" degree in which transitions are allowed only among neighboring points, and the spectrum of the transition matrix decays rapidly. The direct method we present offers several advantages.

(i) The multiscale construction allows efficient approximation of value functions at multiple levels of resolution, where the hierarchy is *automatically* generated. This approach has important applications to hierarchical reinforcement learning (Barto & Mahadevan, 2003), although this is not the explicit focus of this paper.

(ii) It is well-known that the number of iterations necessary for an iterative method to converge can be very large, depending on the condition number of the problem, which in general depends on the number of points, and on the precision required. Increasing precision in the direct inversion technique we propose can be done more efficiently.

(iii) When the state space and the policy are fixed, and many value functions corresponding to different rewards (tasks) need to be computed, iteration schemes do not take advantage of the common structure between the problems. In this case, the number of iterations for finding each solution is multiplied by the number of solutions sought. Our direct inversion technique efficiently encodes the common structure of the state space in the pre-computation step, and then takes advantage of this in the solution of multiple problems, thus enabling transfer across tasks in a completely novel manner.

2. Markov decision processes and the Bellman equation

A finite Markov decision process (MDP) $M = (S, A, P^a_{ss'}, R^a_{ss'})$ is defined by a finite set of states S, a finite set of actions A, a transition model $P^a_{ss'}$ specifying the distribution over future states s' when an action a is performed in state s, and a corresponding reward model $R^a_{ss'}$ specifying a scalar cost or reward (Puterman, 1994). In this paper, the transition probabilities $P^\pi_{ss'}$ defined by a policy π will be represented as a random walk on a weighted, possibly directed, graph G, where $P^\pi_{ss'} = \frac{w^\pi(s,s')}{\sum_t w^\pi(s,t)}$. A value function is a mapping $S \rightarrow \mathcal{R}$ or equivalently a vector in $\mathcal{R}^{|S|}$. Given a policy $\pi : S \rightarrow A$ mapping states to actions, its corresponding value function V^π specifies the expected long-term discounted sum of rewards received

by the agent in any given state s when actions are chosen using the policy. This paper focuses on policy evaluation since this is usually the most expensive step in policy iteration requiring the inversion of the transition matrix. The second phase of policy improvement requires computing the greedy policy, and is of lower complexity. Policy evaluation consists of solving the (Bellman) linear system of equations

$$V^\pi = R + \gamma P^\pi V^\pi$$

where V^π and R are vectors of length $|\mathcal{S}|$, and P^π is a stochastic matrix of size $|\mathcal{S}| \times |\mathcal{S}|$, and $\gamma \in (0, 1]$ is the discount factor. The solution is given by $V^\pi = (I - \gamma P^\pi)^{-1} R$. When $\gamma = 1$ (the non-discounted case), the matrix $(I - P^\pi)^{-1}$ is called the fundamental matrix of the Markov chain P^π, or Green's function, for its interpretation in potential theory and physics (Kemeny et al., 1976).

3. Multiscale analysis of state space with diffusion wavelets

In (Coifman & Maggioni, 2004) ideas and algorithms for natural hierarchical multiscale analysis of Markov chains and graphs are introduced, associated with new families of objects called *diffusion wavelets*. This framework provides a multiscale analysis for functions on a graph through basis functions automatically built on the graph, at different locations and levels of resolution. This allows the efficient analysis, representation, and compression of functions on graphs. Moreover this approach provides a hierarchical compression and organization of the graph itself, including compression of all the powers P^{2^j} of a Markov random walk P, for efficient and accurate computation of behavior of the random walk at all time scales, and efficient computation of several functions of P and its powers, including notably the Green's function $(I - \gamma P)^{-1}$. By "efficient" we mean that the number of operations required is asymptotically (i.e. for fixed precision, and large $|S|$), of order $\mathcal{O}(|S|)$, at least for certain classes of Markov chains P.

We describe how to apply the diffusion wavelet framework to MDPs. We assume the state space can be modeled as a finite, directed or undirected, weighted graph (S, E, W) (our approach generalizes to Riemannian manifolds, which we do not have space to discuss here[1]). If any policy π is executed, it will traverse some subset of the state space $S^\pi \subseteq S$. We will write $x \sim y$ when there is an edge between x and y, and $w(x, y)$ denotes the edge weight. We denote the degree of x by $d(x) = \sum_{x \sim y} w(x, y)$. D represents the diagonal matrix defined by $D(x, x) = d(x)$, and W the matrix defined by $W(x, y) = w(x, y)$. Let P be the natural random walk defined by $P = D^{-1}W$. P is a nonsymmetric matrix, however if the graph is undirected then P is spectrally related to the symmetric self-adjoint operator $D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$ which is called the *normalized graph Laplacian* (Chung, 1997). Note that the weights W can be chosen quite arbitrarily. One choice is geometrical: for example if the state space is sampled from a manifold, the weights can be chosen in a way consistent with the natural random walk on the manifold (Belkin & Niyogi, 2003; Lafon, 2004). Observe that the weights thus obtained are policy independent. Another choice is policy dependent: for example the weights could be chosen so that P is the current policy. Finally, one can interpolate between the two choices above, by assigning weights to the edges both based on their geometric weights and the current policy.

[1]For results related to sampling of Riemannian manifolds and approximation of the Laplacian see (Singer, 2006) and references therein.

3.1. Setup and assumptions

The hypotheses on P are that P^t, $t \geq 0$, should be a Markov diffusion process, in a technical sense made precise in (Coifman & Maggioni, 2004). Qualitatively, this means that P should be *local*, i.e. from every point a random walk takes the agent to only a few nearby points; *smoothing*, i.e. $P^t \delta_x$, for any initial condition δ_x, should be a smooth probability distribution centered about x; *contractive*, i.e. $||P||_2 \leq 1$. For the proposed algorithm to have complexity $\mathcal{O}(|S|)$, two further assumptions are needed: first, the matrix P should be sparse, in the sense that only about $c|S|$ entries are non-zero, where $c > 0$ is a small constant. This is the case for example for the natural random walk on a graph where the vertices have degree bounded by c. Secondly, the eigenvalues $\{\lambda_j\}$ of P should decay rapidly, for example $\#\{j : \lambda_j \geq \epsilon^{2^{-j}}\} \leq c 2^{-j\alpha} \log_2^\alpha(1/\epsilon)$ for some $\alpha > 0$, and some fixed small $\epsilon > 0$. As an example, it is shown in (Coifman & Maggioni, 2004) that this condition is satisfied when P is a discretization of the natural random walk on a smooth compact Riemannian manifold of dimension d, in which case one can choose $\alpha = d/2$.

3.2. Qualitative description

We briefly describe the construction of diffusion wavelets, referring the interested reader to (Coifman & Maggioni, 2004) for further details. The input to the algorithm is a weighted, possibly directed, graph (S, E, W) (which implicitly defines a transition matrix

P) and a "precision" parameter $\epsilon > 0$. For simplicity, we assume here that the graph is undirected, but the construction, as noted in (Coifman & Maggioni, 2004), generalizes to the directed case. In this case the natural random walk on the graph is represented $P = D^{-1}W$, which we assume reversible (this is the case if the graph is undirected). We symmetrize it by conjugation, and take powers to obtain

$$T^t = D^{\frac{1}{2}} P^t D^{-\frac{1}{2}} = (D^{-\frac{1}{2}} W D^{-\frac{1}{2}})^t = (I - \mathcal{L})^t = \sum_{i \geq 0} (1 - \lambda_i)^t \xi_i(\cdot) \xi_i(\cdot)\,. \quad (1)$$

where $\mathcal{L}$ is the normalized Laplacian (Chung, 1997), $\{\lambda_i\}$ and $\{\xi_i\}$ are its eigenvalues and eigenvectors: $\mathcal{L}\xi_i = \lambda_i \xi_i$. Hence the eigenfunctions of T^t are again ξ_i and the i^{th} eigenvalue is $(1-\lambda_i)^t$. A multiresolution decomposition of the functions on the graph is a family of nested subspaces $V_0 \supseteq V_1 \supseteq \cdots \supseteq V_j \supseteq \ldots$ spanned by orthogonal bases of diffusion scaling functions Φ_j. If T^t is interpreted as an operator on functions on the graph, then V_j is defined as the numerical range, up to precision ϵ, of $T^{2^{j+1}-1}$, and the scaling functions are smooth bump functions with some oscillations, at scale roughly 2^{j+1} (measured with respect to geodesic distance). The orthogonal complement of V_{j+1} into V_j is called W_j, and is spanned by a family of orthogonal diffusion wavelets Ψ_j, which are smooth localized oscillatory functions at the same scale.

3.3. A very simple example

We consider the Markov chain on 4 states $\{a, b, c, d\}$:

$$T = \begin{pmatrix} 0.8 & 0.2 & 0 & 0 \\ 0.2 & 0.75 & 0.05 & 0 \\ 0 & 0.05 & 0.75 & 0.2 \\ 0 & 0 & 0.2 & 0.8 \end{pmatrix}.$$

This chain has a "bottleneck" between states $\{a, b\}$ and states $\{c, d\}$. We fix a precision $\epsilon = 10^{-10}$. See Figure 1 for the discussion that follows. The scaling functions Φ_0 are simply $\{\delta_a, \delta_b, \delta_c, \delta_d\}$. We apply T to Φ_0 and orthonormalize to get Φ_1 (Figure 1). Each function in Φ_1 is an "abstract-state", i.e. a linear combination of the original states. We represent T^2 on Φ_1, to get a matrix T_2, apply to Φ_1 and orthonormalize, and so on. At scale 5 we have the basis Φ_5 and the operator T_5, representing T^{2^5} on Φ_5. At the next level, we obtain Φ_6, which is only two dimensional, because $T_5\Phi_5$ has ϵ-rank 2 instead of 4: of the 4 "abstract-states" $T_5\Phi_5$, only two of them are at least ϵ-independent. Observe the two scaling functions in Φ_6 are approximately the asymptotic distribution and the function which distinguishes between the two clusters $\{a, b\}$ and $\{c, d\}$. Then T_6 represents T^{2^6} on Φ_6 and is a 2 by 2 matrix. At scale 10, Φ_{10} is one-dimensional, and is simply the top eigenvector of T (represented in compressed form, on the basis Φ_8), and the matrix T_9 is 1 by 1 and is just the top eigenvalue, 1, of T.

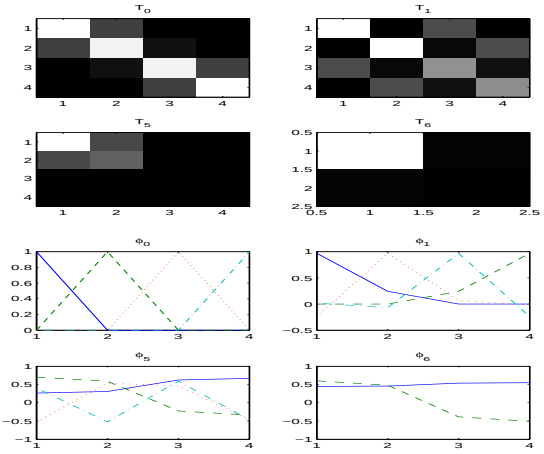

Figure 1. The four panels on the top display matrices representing compressed dyadic powers of T, with gray level representing entry values. The four panels on the bottom illustrate some scaling function bases on the 4-state Markov chain.

Already in this simple example we see that the multiscale analysis generates a sequence of Markov chains, each corresponding to a different time scale (i.e. power of the original Markov chain), represented on a set of scaling functions ("aggregates of states") in compressed form.

3.4. Construction of the scaling functions and wavelets

We sketch the key elements of the construction of the multiscale analysis that are relevant for our purposes, and refer the reader to (Coifman & Maggioni, 2004) for details. The scaling space V_0 at the finest scale is the subspace spanned by $\Phi_0 := \{\delta_x\}_{x \in \mathcal{S}}$, where δ_x is the Dirac delta (evaluation functional) at the point x. Then consider the family $\tilde{\Phi}_0 := \{T^{2^0}\delta_x\}_{x \in \mathcal{S}}$, and let $\tilde{V}_1$ be the span of $\tilde{\Phi}_0$. A careful Gram-Schmidt procedure applied to $\tilde{\Phi}_0$ produces an orthonormal basis of well-localized scaling functions Φ_1, spanning a subspace V_1 close to $\tilde{V}_1$ up to the pre-specified precision ϵ. V_1 is also the subspace spanned by $\{\xi_i : \lambda_i \geq \epsilon\}$, hence in general $\dim V_1 \leq \dim V_0$: the orthonormalization will produce only $\dim V_1$ basis functions in Φ_1. T^2 is represented on the basis Φ_1 via a matrix T_1, and the construction is repeated. By induction, at scale j an orthonormal basis of localized scaling functions Φ_j has been constructed, and T^{2^j} has been represented on this basis via a matrix T_j. Observe that the matrix representing T^{2^j} on this basis has size $|\Phi_j| \times |\Phi_j|$. The

`DiffusionWaveletTree` $(T_0, \Phi_0, J, \epsilon)$:

// T_0: symmetric conjugate to random walk matrix, represented on the basis Φ_0
// Φ_0 : initial basis (usually Dirac's δ-function basis), one function per column
// J : number of levels to compute
// ϵ: precision

for j **from** 0 **to** J **do**,

1. Compute sparse factorization $T_j \sim_\epsilon Q_j R_j$, with Q_j orthogonal.
2. $\Phi_{j+1} \leftarrow Q_j = H_j R_j^{-1}$.
3. $[T_0^{2^j}]_{\Phi_{j+1}}^{\Phi_{j+1}} \sim_{j\epsilon} T_{j+1} \leftarrow R_j R_j^*$.
4. Compute sparse factorization
$$I - \Phi_{j+1}\Phi_{j+1}^* = Q_j' R_j' \,,$$
with Q_j' orthogonal.
5. $\Psi_{j+1} \leftarrow Q_j'$.

end

Figure 2. Pseudo-code for construction of a Diffusion Wavelet Tree

assumptions on the decay of the spectrum of T imply that $|\Phi_j| << |\Phi_0|$ (for $j > 0$), and one says that T^{2^j} is represented in compressed form. For the next stage let $\tilde{\Phi}_j := T^{2^j}\Phi_j$, and V_{j+1} be the span of these vectors. After orthonormalization to get a (smaller) orthonormal set Φ_{j+1} spanning $V_{j+1} \subseteq V_j$. A set of oscillatory functions Ψ_j (the wavelets) spanning W_j, the orthogonal complement of V_{j+1} into V_j, can be constructed similarly. They capture the detail lost from going from V_j to V_{j+1}, and act as high-pass filters in the sense that their expansion in terms of eigenfunctions of the Laplacian ξ_i essentially only involves eigenfunctions corresponding to eigenvalues $\lambda_i \in [\epsilon^{-2^j-1}, \epsilon^{-2^{j+1}-1}]$. In particular their smoothness, is controlled. In Figure 2 this scheme is written in pseudo-code, where the operations above are described in terms of matrices: the Gram-Schmidt procedure at each level orthogonalizes the columns of T_j into the product of an orthogonal matrix Q_j, which forms the basis of scaling functions at the next scale, and a matrix R_j, which represents T_j on Φ_j in the domain and Φ_{j+1} in the range. The assumptions of the process $\{P^t\}$ guarantee that Q_j and R_j can be constructed so that they contain only $\mathcal{O}(|R_j|)$ entries above precision.

3.5. Applications

Diffusion wavelets and wavelet packets have been shown to be an efficient tool for representation and approximation of functions on manifolds and graphs (Coifman & Maggioni, 2004), generalizing to these general spaces the wavelets that have so successfully employed for similar tasks in Euclidean spaces. Applications include the analysis of networks, graphs, document corpora (Coifman & Maggioni, 2004), nonlinear and anisotropic image denoising, learning tasks, and value function approximation (Mahadevan & Maggioni, 2005).

4. Direct multiscale solution of Bellman's equation

In this section we show that the multiscale construction outlined in Section 3 enables a fast direct solution of Bellman's equation.

The starting point are the identities

$$\begin{aligned} V^\pi &= (I - \gamma P^\pi)^{-1} R = \sum_{k\geq 0} (\gamma \Pi^{-\frac{1}{2}} T^\pi \Pi^{\frac{1}{2}})^k R \\ &= \prod_{k\geq 0} (I + \gamma^{2^k} \Pi^{-\frac{1}{2}} (T^\pi)^{2^k} \Pi^{\frac{1}{2}}) R \,, \end{aligned} \tag{2}$$

where $P^\pi = \Pi^{-\frac{1}{2}} T^\pi \Pi^{\frac{1}{2}}$, Π is the matrix whose diagonal is the asymptotic distribution of P, and R is the reward vector. The first identity follows by the definition of T^π, the second is the usual Neumann series expansion for the inverse, and the last identity is called the Schultz formula. The equality holds since each term of the Neumann series appears once and only once in the product, since every integer has a unique binary expansion. Observe that reordering the terms of the summation is allowed because both the sum and the product are absolutely convergent. The formulas hold for $\gamma \leq 1$ and f not in the kernel of $(I - \gamma P^\pi)$. Observe that since $\gamma \leq 1$ and $||P^\pi||_2 \leq 1$, the only case for which this kernel is not trivial is when $\gamma = 1$, and in this case the kernel is the unique (since we assumed the state space is connected) asymptotic distribution of P^π. The sums and products in (2) are of course finite once the precision is fixed.

A key component in the construction of diffusion wavelets was the compression of the (quasi-)dyadic powers of the operator T^π. Since R_i represents the operator $(T^\pi)^{2^i}$ on the basis Φ_i in the domain and Φ_{i+1} in the range, the product $R_j R_{j-1} \cdot \cdots \cdot R_0 f$ is $T^{1+2+2^2+\cdots+2^{j-1}} f = T^{2^j-1} f$, represented on Φ_{j+1}, i.e. "in compressed form". The matrices $[\Phi_{j+1}]_{\Phi_j}^*$ "unpack" this representation back onto the basis Φ_0. In

other words

$$
\begin{aligned}
&[(T^\pi)^{2^j-1} f]_{\Phi_0} \\
&= [\Phi_1]^*_{\Phi_0} \dots [\Phi_{j-1}]^*_{\Phi_{j-2}} [T^{2^j}]^{\Phi_{j-1}}_{\Phi_{j-2}} [T^{2^j}]^{\Phi_{j-2}}_{\Phi_{j-3}} \dots [T]^{\Phi_1}_{\Phi_0} [f]_{\Phi_0} \\
&= [\Phi_1]^*_{\Phi_0} \dots [\Phi_{j-1}]^*_{\Phi_{j-2}} R_j R_{j-1} \dots R_0 f
\end{aligned}
$$

To obtain $T^{2^j} f$ we only need to apply T once more. In this way the computation of $T^{2^j} f$ requires only $\mathcal{O}(j|S|)$ operations, since R_j contains about $\mathcal{O}(|S|)$ entries. This cost should be compared to that of computing directly the matrix T^{2^j}, which is $\mathcal{O}(2^j|S|)$ since this matrix contains about $\mathcal{O}(2^j|S|)$ nonzero entries; this is also the cost of applying about 2^j times the matrix T to f.

Observe that the same Diffusion Wavelet Tree can be used for this multiscale inversion for different values of the discount rate γ.

The method has high-precision, so that it can be the solver for Bellman's equation within a policy-iteration scheme requiring any given precision.

4.1. Comparison with standard direct inversion and iterative methods

The standard direct inversion technique involves the explicit computation of $(I - \gamma P^\pi)^{-1}$. This typically involves the computation of the singular vectors and singular values of $(I - \gamma P^\pi)$: this enables representing $I - \gamma P^\pi = U \Sigma V$ with U, V orthonormal and Σ diagonal, with diagonal entries $\sigma_1 \geq \dots \geq \sigma_{|S|}$. For a fixed precision ϵ, only a partial decomposition $U_N \Sigma_N V_N$ is computed, where N is the largest n for which $\sigma_n \geq ||I - \gamma P^\pi||_2 \epsilon$. We can then write

$$(I - \gamma P^\pi)^{-1} = V^* \Sigma^{-1} U^* .$$

Very stable algorithms are available for the computation of the singular value decomposition. Optimality of the singular vectors with respect to approximation properties of the matrix itself are also well-known and are the main motivation for this technique. Unfortunately these techniques are in general expensive, with complexity $\mathcal{O}(|S|^3)$.

In iterative methods such as value iteration, up to $|S|$ iterations are necessary, and the cost is thus $\mathcal{O}(|S|^2)$. In contrast, our technique has complexity only $\mathcal{O}(|S|)$. However in practice less than $|S|$ iterations may be needed for iterative methods to converge, especially when the problem is well-conditioned (e.g. γ far from 1), and/or low precision in the result is requested. Even when this is the case our method offers several advantages: it generates basis functions tuned to the structure of the problem that efficiently represent the value function, and once computed, this structure allows the direct fast inversion of Bellman's equation for many different rewards R.

We note that the method we propose is different from the wavelet element method for solving integral equations of potential theory: Schultz's formula corresponds to a factorization of the integral operator $(I - \gamma P^\pi)^{-1}$ on a multiscale basis, rather than a standard linear representation of the operator on a basis, which is what wavelet element methods use.

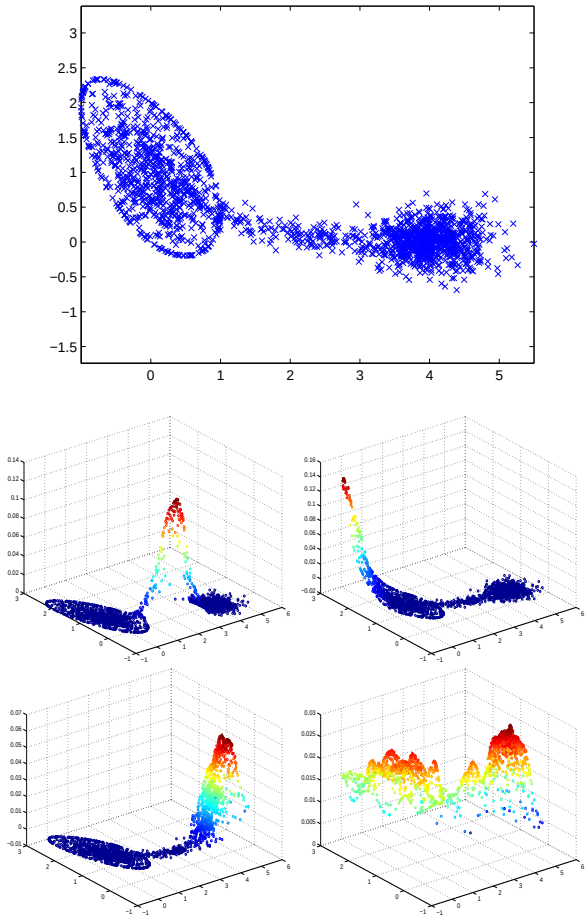

Figure 3. Top: set of samples in a two-room environment. Bottom: four diffusion scaling functions built on the set, at increasing scale. Note the localization at the finer scales, and the global support at coarser scales.

5. Experiments

We constructed the multiscale analysis on several MDPs, on discrete and continuous spaces of different topologies. In fact the technique used is extremely flexible, since it essentially only needs P^π as an input.

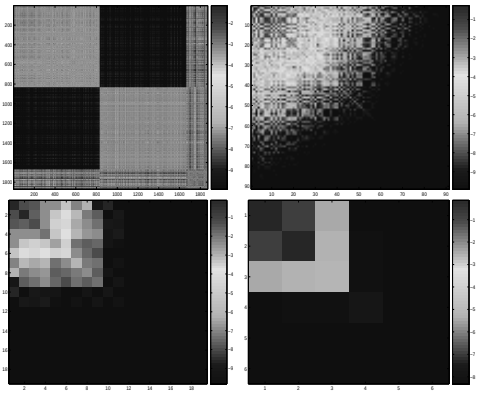

Figure 4. Compression of the powers of the symmetrized random walk T in a continuous two-room environment. From top left to bottom right by rows: T_0, T_1, T_4 and T_6. All the matrices are represented in $\log_{10}$ scale. T_0 is sorted to show the two-room and corridor structures (the algorithm is of course independent of the order of the points): the two large blocks represent transitions within each room, and the bottom-right block are transitions in the corridor, with bands at the bottom and at the right indicating the transitions from the corridor to the rooms. Notice the decreasing size of the matrices. T_6 is very small, and essentially represents only the transition between two states (the two rooms): for time scales of order 2^6 the algorithm has automatically decided this representation is faithful enough for the precision requested.

We describe here one example in some detail. It simulates a continuous two-rooms environment, where the two rooms have an elongated shape and are connected by a corridor. The shape of the rooms and of the corridor is quite arbitrary, the bases are built automatically: we do not require any special topology or shape property for them (except connectedness, without loss of generality): we could have chosen rooms of arbitrary shapes, in arbitrary dimension, as the only input to the algorithm is the set of sampled points (vertices) and the local distances between close-by points (edge weights).

The agent has randomly explored the space, so S consists of $|S|$ randomly scattered points in the rooms (see Figure 3). We construct a natural diffusion associated with the random walk in the two rooms, restricted to the states S actually explored, by letting $W(i,j) = e^{-2||x_i - x_j||^2}$. This diffusion approximates the natural random walk (Brownian motion) in the continuous domain (Belkin & Niyogi, 2003; Lafon, 2004), corresponding to a policy of random moves. We then construct the corresponding multiscale analysis, with precision set to 10^{-10}. In Figure 3 we represent some of the scaling functions we obtain. In Figure 4 we represent compressed dyadic powers of this random walk. In Figure 5, left, we compare the direct computation time of $(I - \gamma P^\pi)^{-1}$ and the computation time for the multiscale structure, i.e. the pre-processing time for the two direct methods under consideration. We then pick a random reward R on S (a vector of white Gaussian noise), and compute the corresponding value function in three different ways:

(i) direct computation of the matrix $I - \gamma P^\pi$,

(ii) Schultz's method and diffusion wavelet transform as in (2),

(iii) conjugate gradient descent for symmetric matrices.

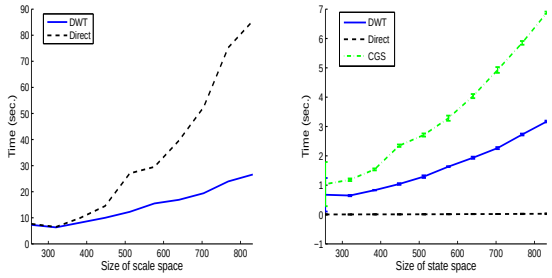

Figure 5. Left: mean and standard deviation of running time for solving a Bellman equation on a random walk in the two-room environment, as a function of the number of states explored (x-axis). We compared direct DWT inversion, iterative Conjugate Gradient Squared method (Matlab implementation) and direct inversion. Left: preprocessing time, comparing computation of the full inverse and construction diffusion wavelet tree. Right: computation time of applying the inversion scheme, comparing direct inverse, Schultz's method with diffusion wavelet transform, and symmetric conjugate gradient.

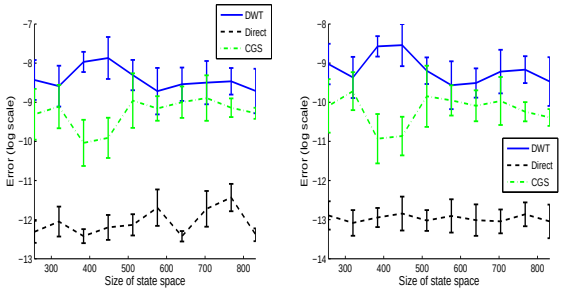

Figure 6. Precision, defined as $\log_{10}$ of the Bellman residual error $||(I - \gamma P^\pi)\tilde{V}^\pi - R||_p$, where $\tilde{V}^\pi$ is the computed solution, achieved by the different methods. The precision requested was $1e-10$. We show the results for $p = 2$ (left) and $p = \infty$ (right).

In this example we set $\gamma = 1$. We repeat the above for $|S| = 320, 400, 480, 560, 640, 720, 800, 880, 960, 1040$ and, for each S, for 10 randomly generated rewards R. The first two methods are direct: we look at both

the pre-processing time for computing, respectively, the inverse matrix and the diffusion wavelet tree (see Figure 5 left). Then we compare, over several random choices of the reward vector, the mean and standard deviation of the time for computing the corresponding value function, with all three methods: see Figure 5, right. Finally, in Figure 6 we show the L^2- and L^∞-norms of the Bellman residual $((I - P^\pi)\tilde{V}^\pi - R$, where $\tilde{V}^\pi$ is the estimated value function), achieved by the three methods.

We stress that the code that implements the construction of Diffusion Wavelet Tree and Schultz's formula is written mainly in Matlab and large parts of it are not optimized in any way; on the contrary the codes distributed with Matlab for conjugate gradient and direct inversion have been highly optimized. The results here are thus qualitative and not absolute, but point out that at the very least the direct solution using diffusion wavelets seems competitive with state-of-the-art methods, even before having optimized the code implementing it. Much future work will be devoted for these optimizations, which will result in speed-ups and the ability to tackle larger and larger problems. In particular factored spaces can be tackled by using factored diffusion wavelets.

6. Conclusions and Future Work

We applied a novel framework (Coifman & Maggioni, 2004) based on multiscale analysis of graphs and Markov diffusion processes to designing a new fast policy evaluation algorithm. The approach constructs a hierarchical set of diffusion wavelet basis functions for efficiently representing powers of the transition matrix. Many directions for extending this approach are being studied, including applications to policy iteration and hierarchical reinforcement learning. For large or continuous state spaces, where graphs represent a sample of the underlying state space, Nyström approximations (Fowlkes et al., 2001) or more refined interpolation techniques, such as the multiscale extension of (Coifman & Maggioni, 2004) can be exploited to interpolate basis functions to novel points. Extensions to factored state spaces are also being investigated.

6.1. Continuous Domains

In continuous domains, a critical ingredient for success of a method based on the representation of the value function on some basis of the state-action space is the capability of approximating the value function efficiently in this basis, and in extending these basis functions to novel states. Diffusion wavelets approximate efficiently various important classes of functions, such as piecewise smooth functions, and can be extended in a multiscale fashion (Coifman & Maggioni, 2004).

Acknowledgements: This research is supported in part by grants from the NSF DMS-0512050 and IIS-0534999.

References

Barto, A., & Mahadevan, S. (2003). Recent advances in hierarchical reinforcement learning. *Discrete Event Systems Journal*, *13*, 41–77.

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, *15*, 1373–1396.

Chung, F. (1997). *Spectral Graph Theory*. American Mathematical Society.

Coifman, R. R., & Maggioni, M. (2004). Diffusion wavelets. *Tech. Rep. YALE/DCS/TR-1303, Yale Univ., Appl. Comp. Harm. Anal.* To appear.

Fowlkes, C., Belongie, S., & Malik, J. (2001). Efficient spatiotemporal grouping using the nyström method. *CVPR*.

Greengard, L., & Rokhlin, V. (1987). A fast algorithm for particle simulations. *J Comput Phys*, *73*, 325–348.

Kemeny, J., Snell, J., & Knapp, A. (1976). *Denumerable markov chains*. Springer-Verlag.

Lafon, S. (2004). *Diffusion maps and geometric harmonics*. Doctoral dissertation, Yale University, Dept of Mathematics & Applied Mathematics.

Lagoudakis, M., & Parr, R. (2003). Least-squares policy iteration. *Journal of Machine Learning Research*, *4*, 1107–1149.

Mahadevan, S. (2005). Representation policy iteration. *Proceedings of the* 21^{st} *International Conference on Uncertainty in Artificial Intelligence*.

Mahadevan, S., & Maggioni, M. (2005). Value function approximation with diffusion wavelets and laplacian eigenfunctions. *University of Massachusetts, Department of Computer Science Technical Report TR-2005-38; Proc. NIPS 2005*.

Puterman, M. L. (1994). *Markov decision processes*. New York, USA: Wiley Interscience.

Singer, A. (2006). From graph to manifold Laplacian: the convergence rate. *Appl. Comp. Harm. Anal., to appear*.