Incremental Basis Construction from Temporal Difference Error

Yi Sun, Faustino Gomez, Mark Ring, Jürgen Schmidhuber

IDSIA, USI & SUPSI, Switzerland

June 2011
A Markov Reward Process (MRP) is defined by the 4-tuple \( (S, P, r, \gamma) \):

- \( S \) is the state space
- \( P \) is an \( S \times S \) transition matrix with \( P_{i,j} = \Pr[s_{t+1} = j | s_t = i] \)
- \( r \) is the reward function
- \( \gamma \in [0, 1) \) is the discount factor

The Value Function, \( v \in \mathbb{R}^S \), is the solution of the Bellman equation:

\[
v(s) = r(s) + \gamma \mathbb{E}_P[v(s')]
\]

Let \( L = I - \gamma P \), then

\[
v(s) = L^{-1}r(s)
\]
A Markov Reward Process (MRP) is defined by the 4-tuple $\langle S, P, r, \gamma \rangle$. 

- $S = \{1, \ldots \}$ is the state space.
- $P$ is an $S \times S$ transition matrix with $P_{i,j} = \Pr[s_{t+1} = j | s_t = i]$.
- $r \in \mathbb{R}^S$ is the reward function.
- $\gamma \in [0, 1)$ is the discount factor.

The Value Function $v \in \mathbb{R}^S$ is the solution of the Bellman equation $v = r + \gamma P v$. Let $L = I - \gamma P$, then $v = L^{-1} r$. 

Sun, Gomez, Ring, Schmidhuber (IDSIA)
A Markov Reward Process (MRP) is defined by the 4-tuple \( \langle S, P, r, \gamma \rangle \)

- \( S = \{1, \ldots, S\} \) is the state space
A Markov Reward Process (MRP) is defined by the 4-tuple $\langle S, P, r, \gamma \rangle$

- $S = \{1, \ldots, S\}$ is the state space
- $P$ is an $S \times S$ transition matrix with $\{P\}_{i,j} = \Pr[s_{t+1} = j \mid s_t = i]$
- $r \in \mathbb{R}^S$ is the reward function
- $\gamma \in [0, 1)$ is the discount factor

The Value Function, $v \in \mathbb{R}^S$, is the solution of the Bellman equation $v = r + \gamma P v$. Let $L = I - \gamma P$, then $v = L^{-1}r$. 

Sun, Gomez, Ring, Schmidhuber (IDSIA)
A Markov Reward Process (MRP) is defined by the 4-tuple \( \langle S, P, r, \gamma \rangle \)

- \( S = \{1, \ldots, S\} \) is the state space
- \( P \) is an \( S \times S \) transition matrix with \( \{P\}_{i,j} = \Pr[s_{t+1} = j \mid s_t = i] \)
- \( r \in \mathbb{R}^S \) is the reward function
- \( \gamma \in [0, 1) \) is the discount factor

The Value Function, \( v \in \mathbb{R}^S \), is the solution of the Bellman equation

\[
v = r + \gamma P v.
\]

Let \( L = I - \gamma P \), then

\[
v = L^{-1} r.
\]
A Markov Reward Process (MRP) is defined by the 4-tuple \( \langle S, P, r, \gamma \rangle \)

- \( S = \{1, \ldots, S\} \) is the state space
- \( P \) is an \( S \times S \) transition matrix with \( \{P\}_{i,j} = \Pr[s_{t+1} = j \mid s_t = i] \)
- \( r \in \mathbb{R}^S \) is the reward function
- \( \gamma \in [0, 1) \) is the discount factor
A Markov Reward Process (MRP) is defined by the 4-tuple \( \langle S, P, r, \gamma \rangle \)

- \( S = \{1, \ldots, S\} \) is the state space
- \( P \) is an \( S \times S \) transition matrix with \( \{P\}_{i,j} = \Pr[s_{t+1} = j \mid s_t = i] \)
- \( r \in \mathbb{R}^S \) is the reward function
- \( \gamma \in [0, 1) \) is the discount factor

The Value Function, \( \nu \in \mathbb{R}^S \), is the solution of the Bellman equation

\[
\nu = r + \gamma P \nu.
\]
A *Markov Reward Process* (MRP) is defined by the 4-tuple $\langle S, P, r, \gamma \rangle$

- $S = \{1, \ldots, S\}$ is the state space
- $P$ is an $S \times S$ transition matrix with $\{P\}_{i,j} = \Pr[s_{t+1} = j \mid s_t = i]$
- $r \in \mathbb{R}^S$ is the reward function
- $\gamma \in [0, 1)$ is the discount factor

The *Value Function*, $\nu \in \mathbb{R}^S$, is the solution of the Bellman equation

$$\nu = r + \gamma P\nu.$$

- Let $L = I - \gamma P$, then $\nu = L^{-1} r$
Linear function approximation (LFA):
\[ \hat{v} = \Phi \theta, \]
where \( \Phi = [\phi_1, \ldots, \phi_N] \) are \( N \) basis functions, and \( \theta = [\theta_1, \ldots, \theta_N] \) are the weights.

The Bellman Error \( \varepsilon \in \mathbb{R}^S \) is defined as
\[ \varepsilon = r + \gamma P \hat{v} - \hat{v} = r - L \Phi \theta. \]

\( \varepsilon \equiv 0 \iff v \equiv \Phi \theta \)

\( \varepsilon \) is the expectation of the TD error.
Linear function approximation (LFA): \( \hat{v} = \Phi \theta \), where

\( \Phi = [\phi_1, \ldots, \phi_N] \) are \( N \) \((\mathbb{N} / \text{uni226A})\) basis functions

\( \theta = [\theta_1, \ldots, \theta_N] / \text{uni22BA} \) are the weights

The Bellman Error \( \epsilon \in \mathbb{R}^S \) is defined as

\[
\epsilon = r + \gamma \hat{v} - \hat{v} = r - \mathbb{E}_\Phi \theta.
\]
Linear function approximation (LFA): $\hat{v} = \Phi \theta$, where

- $\Phi = [\phi_1, \ldots, \phi_N]$ are $N$ ($N \ll S$) basis functions

$\epsilon$ is the expectation of the TD error

$\epsilon \equiv 0 \iff v \equiv \Phi \theta$
Linear function approximation (LFA): \( \hat{v} = \Phi \theta \), where

- \( \Phi = [\phi_1, \ldots, \phi_N] \) are \( N \) (\( N \ll S \)) basis functions
- \( \theta = [\theta_1, \ldots, \theta_N]^T \) are the weights
Linear function approximation (LFA): $\hat{v} = \Phi \theta$, where

- $\Phi = [\phi_1, \ldots, \phi_N]$ are $N$ ($N \ll S$) basis functions
- $\theta = [\theta_1, \ldots, \theta_N]^T$ are the weights

The Bellman Error $\varepsilon \in \mathbb{R}^S$ is defined as

$$\varepsilon = r + \gamma P \hat{v} - \hat{v} = r - L\Phi \theta.$$
Linear function approximation (LFA): \( \hat{v} = \Phi \theta \), where

- \( \Phi = [\phi_1, \ldots, \phi_N] \) are \( N \) \((N \ll S)\) basis functions
- \( \theta = [\theta_1, \ldots, \theta_N]^{\top} \) are the weights

The Bellman Error \( \varepsilon \in \mathbb{R}^S \) is defined as

\[
\varepsilon = r + \gamma P \hat{v} - \hat{v} = r - L \Phi \theta.
\]

- \( \varepsilon \equiv 0 \iff v \equiv \Phi \theta \)
**Linear function approximation (LFA):** $\hat{v} = \Phi \theta$, where

- $\Phi = [\phi_1, \ldots, \phi_N]$ are $N$ ($N \ll S$) basis functions
- $\theta = [\theta_1, \ldots, \theta_N]^\top$ are the weights

The **Bellman Error** $\varepsilon \in \mathbb{R}^S$ is defined as

$$\varepsilon = r + \gamma P \hat{v} - \hat{v} = r - L \Phi \theta.$$ 

- $\varepsilon \equiv 0 \iff v \equiv \Phi \theta$
- $\varepsilon$ is the expectation of the TD error
Preliminary

The LFA \( \hat{v} = \Phi \theta \) depends on both \( \theta \) and \( \Phi \).

To find \( \theta \):

TD (Sutton, 1988), LSTD (Bradtke et al., 1996), etc.

To construct \( \Phi \):

Bellman error basis functions (BEBFs, Wu and Givan, 2005; Keller et al. 2006; Parr et al. 2007; Mahadevan and Liu 2010)
Proto-value basis functions (Mahadevan et al., 2006)
Reduced-rank predictive state representations (Boots and Gordon, 2010)
L1-regularized feature selection (Kolter and Ng, 2009)
The LFA $\hat{v} = \Phi \theta$ depends on both $\theta$ and $\Phi$. 

To find $\theta$: TD (Sutton, 1988), LSTD (Bradtke et al., 1996), etc.

To construct $\Phi$: Bellman error basis functions (BEBFs, Wu and Givan, 2005; Keller et al. 2006; Parr et al. 2007; Mahadevan and Liu 2010)
Proto-value basis functions (Mahadevan et al., 2006)
Reduced-rank predictive state representations (Boots and Gordon, 2010)
L1-regularized feature selection (Kolter and Ng, 2009)
The LFA $\hat{v} = \Phi \theta$ depends on both $\theta$ and $\Phi$.

- To find $\theta$:

  - TD (Sutton, 1988), LSTD (Bradtke et al., 1996), etc.
  - Bellman error basis functions (BEBFs, Wu and Givan, 2005; Keller et al. 2006; Parr et al. 2007; Mahadevan and Liu 2010)
  - Proto-value basis functions (Mahadevan et al., 2006)
  - Reduced-rank predictive state representations (Boots and Gordon, 2010)
  - L1-regularized feature selection (Kolter and Ng, 2009)
The LFA $\hat{v} = \Phi \theta$ depends on both $\theta$ and $\Phi$.

- To find $\theta$:
  - TD (Sutton, 1988), LSTD (Bradtke et al., 1996), etc.
The LFA $\hat{\nu} = \Phi \theta$ depends on both $\theta$ and $\Phi$.

- To find $\theta$:
  - TD (Sutton, 1988), LSTD (Bradtke et al., 1996), etc.

- To construct $\Phi$:
The LFA $\hat{v} = \Phi \theta$ depends on both $\theta$ and $\Phi$.

- To find $\theta$:
  - TD (Sutton, 1988), LSTD (Bradtke et al., 1996), etc.

- To construct $\Phi$:
  - Bellman error basis functions (BEBFs, Wu and Givan, 2005; Keller et al. 2006; Parr et al. 2007; Mahadevan and Liu 2010)
The LFA $\hat{\nu} = \Phi \theta$ depends on both $\theta$ and $\Phi$.

- **To find $\theta$:**
  - TD (Sutton, 1988), LSTD (Bradtke et al., 1996), etc.

- **To construct $\Phi$:**
  - Bellman error basis functions (BEBFs, Wu and Givan, 2005; Keller et al. 2006; Parr et al. 2007; Mahadevan and Liu 2010)
  - Proto-value basis functions (Mahadevan et al., 2006)
The LFA $\hat{\nu} = \Phi \theta$ depends on both $\theta$ and $\Phi$.

- **To find $\theta$:**
  - TD (Sutton, 1988), LSTD (Bradtke et al., 1996), etc.

- **To construct $\Phi$:**
  - Bellman error basis functions (BEBFs, Wu and Givan, 2005; Keller et al. 2006; Parr et al. 2007; Mahadevan and Liu 2010)
  - Proto-value basis functions (Mahadevan et al., 2006)
  - Reduced-rank predictive state representations (Boots and Gordon, 2010)
The LFA $\hat{\nu} = \Phi \theta$ depends on both $\theta$ and $\Phi$. 

- **To find $\theta$:**
  - TD (Sutton, 1988), LSTD (Bradtke et al., 1996), etc.

- **To construct $\Phi$:**
  - Bellman error basis functions (BEBFs, Wu and Givan, 2005; Keller et al. 2006; Parr et al. 2007; Mahadevan and Liu 2010)
  - Proto-value basis functions (Mahadevan et al., 2006)
  - Reduced-rank predictive state representations (Boots and Gordon, 2010)
  - L1-regularized feature selection (Kolter and Ng, 2009)
Intuition: "Bellman error, loosely speaking, points towards the optimal value function," (Parr et al., 2007)

Construction:

\[
\phi(k) = r
\]

At stage \( k > 1 \)

Compute TD fixpoint \( \theta(k) \) w.r.t the \( k \) current basis function \( \Phi(k) \)

Get the Bellman error \( \varepsilon(k) = r - L \Phi(k) \theta(k) \)

Expand:

\[
\Phi(k+1) = \varepsilon(k) \Phi(k) \ldots
\]

Sequences of BEBFs form orthogonal basis (Parr et al. 2007)

In sufficient number, any value function can be represented.
Bellman Error Basis Functions

Intuition: "Bellman error, loosely speaking, points towards the optimal value function", (Parr et al., 2007)
Bellman Error Basis Functions

Intuition: "Bellman error, loosely speaking, point[s] towards the optimal value function", (Parr et al., 2007)

Construction:
Bellman Error Basis Functions

Intuition: "Bellman error, loosely speaking, point[s] towards the optimal value function", (Parr et al., 2007)

Construction:

\[ \phi^{(1)} = r \]
Intuition: ”Bellman error, loosely speaking, point[s] towards the optimal value function”, (Parr et al., 2007)

Construction:
- $\phi^{(1)} = r$
- At stage $k > 1$
Bellman Error Basis Functions

Intuition: "Bellman error, loosely speaking, point[s] towards the optimal value function", (Parr et al., 2007)

Construction:

- $\phi^{(1)} = r$
- At stage $k > 1$
  - Compute TD fixpoint $\theta^{(k)}$ w.r.t the $k$ current basis function $\Phi^{(k)}$
Bellman Error Basis Functions

Intuition: ”Bellman error, loosely speaking, point[s] towards the optimal value function”, (Parr et al., 2007)

Construction:

- $\phi^{(1)} = r$
- At stage $k > 1$
  - Compute TD fixpoint $\theta^{(k)}$ w.r.t the $k$ current basis function $\Phi^{(k)}$
  - Get the Bellman error $\varepsilon^{(k)} = r - L\Phi^{(k)}\theta^{(k)}$
Bellman Error Basis Functions

Intuition: "Bellman error, loosely speaking, point[s] towards the optimal value function", (Parr et al., 2007)

Construction:

- $\phi^{(1)} = r$
- At stage $k > 1$
  - Compute TD fixpoint $\theta^{(k)}$ w.r.t the $k$ current basis function $\Phi^{(k)}$
  - Get the Bellman error $\varepsilon^{(k)} = r - L\Phi^{(k)}\theta^{(k)}$
  - Expand: $\Phi^{(k+1)} = [\Phi^{(k)} : \varepsilon^{(k)}]$. 
Bellman Error Basis Functions

Intuition: "Bellman error, loosely speaking, point[s] towards the optimal value function", (Parr et al., 2007)

Construction:
- \( \phi^{(1)} = r \)
- At stage \( k > 1 \)
  - Compute TD fixpoint \( \theta^{(k)} \) w.r.t. the \( k \) current basis function \( \Phi^{(k)} \)
  - Get the Bellman error \( \varepsilon^{(k)} = r - L\Phi^{(k)}\theta^{(k)} \)
  - Expand: \( \Phi^{(k+1)} = [\Phi^{(k)} : \varepsilon^{(k)}] \).

Sequences of BEBFs form orthogonal basis (Parr et al. 2007)
Intuition: "Bellman error, loosely speaking, points towards the optimal value function", (Parr et al., 2007)

Construction:

- $\phi^{(1)} = r$
- At stage $k > 1$
  - Compute TD fixpoint $\theta^{(k)}$ w.r.t the $k$ current basis function $\Phi^{(k)}$
  - Get the Bellman error $\varepsilon^{(k)} = r - L\Phi^{(k)}\theta^{(k)}$
  - Expand: $\Phi^{(k+1)} = [\Phi^{(k)} : \varepsilon^{(k)}]$.

Sequences of BEBFs form orthogonal basis (Parr et al. 2007)

- In sufficient number, any value function can be represented
Problem with BEBF
Slow convergence when $\gamma \to 1$.
Reason: failed to take into account the transition structure

Theorem
Let $\hat{J}(k)$ and $\hat{J}(k+1)$ be the squared value error corresponding to the BEBF basis functions $\Phi(k)$ and $\Phi(k+1)$. Then

$$\rho(k) = \frac{\hat{J}(k+1)}{\hat{J}(k)} \leq \gamma^2.$$
Problem with BEBF
Problem with BEBF

- Slow convergence when $\gamma \rightarrow 1$. 

Theorem
Let $\hat{J}(k)$ and $\hat{J}(k+1)$ be the squared value error corresponding to the BEBF basis functions $\Phi(k)$ and $\Phi(k+1)$. Then

$$\rho(k) = \hat{J}(k+1) \hat{J}(k) \leq \gamma^2.$$
Problem with BEBF

- Slow convergence when $\gamma \to 1$.
- Reason: failed to take into account the transition structure
Problem with BEBF

- Slow convergence when $\gamma \to 1$.
- Reason: failed to take into account the transition structure

**Theorem**

Let $\hat{J}^{(k)}$ and $\hat{J}^{(k+1)}$ be the squared value error corresponding to the BEBF basis functions $\Phi^{(k)}$ and $\Phi^{(k+1)}$. Then

$$\rho^{(k)} = \frac{\hat{J}^{(k+1)}}{\hat{J}^{(k)}} \leq \gamma^2.$$
A Simple Example

- \( P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \).

- \( r \in \mathbb{R}^2 \) moves along the unit square

- Start from empty basis set
  - The first BEBF is the reward.

- Distance between the curve and the origin denotes \((\rho_{(1)}^{(1/2)})\)
Fix \( \hat{v} = \Phi \theta \) as the current value function estimation, then adding \( \phi = v - \hat{v} \) with weight 1 eliminated the error completely. Simple derivation gives:

\[
\phi = v - \Phi \theta = L - r - L \Phi \theta = L - \left( r - \Phi \theta \right) = L - \epsilon.
\]

Observe: \( \phi \) is the solution to the Bellman equation:

\[
\phi = \epsilon + \gamma P \phi
\]

\( \phi \) is the value function of the Bellman error (V-BEBF). \( \phi \) can be estimated by any RL algorithm, with TD error as the reward.
Fix $\hat{v} = \Phi \theta$ as the current value function estimation, then
Fix $\hat{v} = \Phi \theta$ as the current value function estimation, then

- Adding $\phi = v - \hat{v}$ with weight 1 eliminated the error completely
V-BEBF: Main Idea

Fix $\hat{v} = \Phi \theta$ as the current value function estimation, then

- Adding $\phi = v - \hat{v}$ with weight 1 eliminated the error completely
- Simple derivation gives

$$\phi = v - \Phi \theta$$

$$= L^- r - L^- L \Phi \theta$$

$$= L^- (r - \Phi \theta)$$

$$= L^- \epsilon.$$
Fix $\hat{v} = \Phi \theta$ as the current value function estimation, then

- Adding $\phi = v - \hat{v}$ with weight 1 eliminated the error *completely*
- Simple derivation gives

$$
\phi = v - \Phi \theta \\
= L^{-} r - L^{-} L \Phi \theta \\
= L^{-} (r - \Phi \theta) \\
= L^{-} \varepsilon.
$$

Observe: $\phi$ is the solution to the Bellman equation $\phi = \varepsilon + \gamma P \phi$
Fix $\hat{v} = \Phi \theta$ as the current value function estimation, then

- Adding $\phi = v - \hat{v}$ with weight 1 eliminated the error completely
- Simple derivation gives

$$\phi = v - \Phi \theta$$
$$= L^\top r - L^\top L \Phi \theta$$
$$= L^\top (r - \Phi \theta)$$
$$= L^\top \varepsilon.$$

Observe: $\phi$ is the solution to the Bellman equation $\phi = \varepsilon + \gamma P \phi$

- $\phi$ is the value function of the Bellman error (V-BEBF)
Fix $\hat{v} = \Phi \theta$ as the current value function estimation, then

- Adding $\phi = v - \hat{v}$ with weight 1 eliminated the error *completely*
- Simple derivation gives

$$\phi = v - \Phi \theta$$

$$= L^{-1} r - L^{-1} L \Phi \theta$$

$$= L^{-1} (r - \Phi \theta)$$

$$= L^{-1} \varepsilon.$$

Observe: $\phi$ is the solution to the Bellman equation $\phi = \epsilon + \gamma P \phi$

- $\phi$ is the *value function of the Bellman error* (V-BEBF)
- $\phi$ can be estimated by any RL algorithm, with TD error as the reward
V-BEBF: Comparison to BEBF

Both are reward sensitive, using Bellman error. When computed exactly, representing a value function may require a long sequence of BEBFs, but a single V-BEBF is enough. When approximated, the sequence of V-BEBFs converges much faster than BEBFs, when $\gamma \rightarrow 1$. 

Sun, Gomez, Ring, Schmidhuber (IDSIA)
V-BEBF: Comparison to BEBF

- Both are reward sensitive, using Bellman error
V-BEBF: Comparison to BEBF

- Both are reward sensitive, using Bellman error

- When computed exactly, representing a value function may require a long sequence of BEBFs, but a single V-BEBF is enough.
Both are reward sensitive, using Bellman error

When computed exactly, representing a value function may require a long sequence of BEBFs, but a single V-BEBF is enough.

When approximated, the sequence of V-BEBFs converges much faster than BEBFs, when \( \gamma \to 1 \).
V-BEBF: Framework

V-BEBF suggests a natural way to organize RL learners in hierarchy. A primary learner builds an estimation upon a set of basis functions and propagates the TD-error to a secondary learner. The secondary learner estimates the value function of the TD-error, which then becomes the new basis function used by the primary learner.

Sun, Gomez, Ring, Schmidhuber (IDSIA)
V-BEBF: Framework

V-BEBF suggests a natural way to organize RL learners in hierarchy
V-BEBF suggests a natural way to organize RL learners in hierarchy

- A *primary* learner builds the estimation upon a set of basis functions, and propagates the TD-error to a secondary learner.
V-BEBF: Framework

V-BEBF suggests a natural way to organize RL learners in hierarchy

- A *primary* learner builds the estimation upon a set of basis functions and propagates the TD-error to a secondary learner.

- The *secondary* learner estimates the value function of the TD-error, which then becomes the new basis function used by the primary learner.
We are given a set of $M$ raw basis functions $\Psi = [\psi_1, \ldots, \psi_M]$. From $\Psi$ we construct $N$ refined basis functions through linear mapping:

$$\Phi = [\phi_1, \ldots, \phi_N] = \Psi [w_1, \ldots, w_N].$$
We are given a set of $M$ raw basis functions $\Psi = [\psi_1, \ldots, \psi_M]$
Incremental Basis Projection

We are given a set of $M$ raw basis functions $\Psi = [\psi_1, \ldots, \psi_M]$

From $\Psi$ we construct $N$ refined basis functions through linear mapping:

$$\Phi = [\phi_1, \ldots, \phi_N] = \Psi[w_1, \ldots, w_N].$$
Incremental Basis Projection

We are given a set of $M$ raw basis functions $\Psi = [\psi_1, \ldots, \psi_M]$

From $\Psi$ we construct $N$ refined basis functions through linear mapping:

$$\Phi = [\phi_1, \ldots, \phi_N] = \Psi [w_1, \ldots, w_N].$$

IBP: Construct one $w_k$ at stage $k$
We are given a set of $M$ raw basis functions $\Psi = [\psi_1, \ldots, \psi_M]$

From $\Psi$ we construct $N$ refined basis functions through linear mapping:

$$\Phi = [\phi_1, \ldots, \phi_N] = \Psi [w_1, \ldots, w_N].$$

IBP: Construct one $w_k$ at stage $k$
We are given a set of $M$ raw basis functions $\Psi = [\psi_1, \ldots, \psi_M]$. From $\Psi$ we construct $N$ refined basis functions through linear mapping:

$$\Phi = [\phi_1, \ldots, \phi_N] = \Psi [w_1, \ldots, w_N].$$

IBP: Construct one $w_k$ at stage $k$. 

\[
\begin{align*}
\nu &= \Phi \theta = \Psi W \
\end{align*}
\]
Incremental Basis Projection

If the value function is a linear combination of refined basis functions, it is also a linear combination of raw basis functions. So why?

- Small number of basis functions $\Rightarrow$ Fast convergence
- Small number of basis functions $\Rightarrow$ High estimation accuracy

Therefore, it only affects the estimation indirectly.
If the value function is linear combination of refined basis functions, it is also linear combination of raw basis functions. So Why?
If the value function is linear combination of refined basis functions, it is also linear combination of raw basis functions. So Why?

- Small number of basis functions \(\implies\) Fast convergence
If the value function is linear combination of refined basis functions, it is also linear combination of raw basis functions. So Why?

- Small number of basis functions $\implies$ Fast convergence
- Small number of basis functions $\implies$ High estimation accuracy
If the value function is linear combination of refined basis functions, it is also linear combination of raw basis functions. So Why?

- Small number of basis functions $\implies$ Fast convergence
- Small number of basis functions $\implies$ High estimation accuracy
  - Only the learner of the refined basis functions works on raw basis functions
If the value function is linear combination of refined basis functions, it is also linear combination of raw basis functions. So Why?

- Small number of basis functions $\Rightarrow$ Fast convergence
- Small number of basis functions $\Rightarrow$ High estimation accuracy
  - Only the learner of the refined basis functions works on raw basis functions
  - Therefore it only affect the estimation indirectly
IBP with V-BEBF

Approximate each column \( w \) so that \( \Psi_w \) approximates V-BEBF.

Sparsity constraints on \( w \) to make the computation tractable.

Each refined basis function depends only on a handful of raw basis functions.

In this work we simply choose \( B/\text{uni} \) \( M \) entries in \( w \) at random.

Combine with LSTD to attain batch version (\( O(M^3/\text{slash.left}^2) \) in time, \( O(M) \) in storage), with TD to attain online version (\( O(MB) \)).
Approximate each column $w_k$ so that $\Psi w_k$ approximates V-BEBF.
Approximate each column $w_k$ so that $\Psi w_k$ approximates V-BEBF.

Sparsity constraints on $w_k$ to make the computation tractable.
Approximate each column $w_k$ so that $\Psi w_k$ approximates V-BEBF

Sparsity constraints on $w_k$ to make the computation tractable

- Each refined basis function depends only on a handful of raw basis functions
IBP with V-BEBF

Approximate each column $w_k$ so that $\Psi w_k$ approximates V-BEBF

Sparsity constraints on $w_k$ to make the computation tractable

- Each refined basis function depends only on a handful of raw basis functions
- In this work we simply choose $B \ll M$ entries in $w_n$ at random.
Approximate each column $w_k$ so that $\Psi w_k$ approximates V-BEBF

Sparsity constraints on $w_k$ to make the computation tractable

- Each refined basis function depends only on a handful of raw basis functions
- In this work we simply choose $B \ll M$ entries in $w_n$ at random.

Combine with LSTD to attain batch version ($O(M^{3/2})$ in time, $O(M)$ in storage), with TD to attain online version ($O(MB)$).
Randomly generated MRP, 500 states, branching factor 5

Randomly generated binary raw basis functions (30% non-zero)

Error measured in mean-square value error w.r.t. LSTD solution.

In batch case, $B = N = \sqrt{M}$, the training trajectory length is 5000.
Online

$M=1000$

$M=200$

$\gamma=0.99$

$\gamma=0.999$

$\gamma=0.999$

$\gamma=0.99$

IBP!V

IBP!B

TD

IBP!V!cor

$\gamma=0.99$

$\gamma=0.999$

$\gamma=0.999$

$\gamma=0.99$
Simple method for incrementally building up basis functions — Just use the value function of the Bellman error

Rather effective compare to BEBF when \( \gamma \to 1 \)

Extensions:
- Deeper hierarchy
- Multiple secondary learners
- Incorporating memory for the secondary learner

Sun, Gomez, Ring, Schmidhuber (IDSIA)
Conclusion

- Simple method for incrementally building up basis functions — Just use the value function of the Bellman error
Conclusion

- Simple method for incrementally building up basis functions — Just use the value function of the Bellman error

- Rather effective compare to BEBF when $\gamma \rightarrow 1$
Conclusion

- Simple method for incrementally building up basis functions — Just use the value function of the Bellman error
- Rather effective compare to BEBF when $\gamma \to 1$
- Extensions:
Conclusion

- Simple method for incrementally building up basis functions — Just use the value function of the Bellman error
- Rather effective compare to BEBF when $\gamma \to 1$
- Extensions:
  - Deeper hierarchy
Conclusion

- Simple method for incrementally building up basis functions — Just use the value function of the Bellman error

- Rather effective compared to BEBF when $\gamma \to 1$

- Extensions:
  - Deeper hierarchy
  - Multiple secondary learners
Conclusion

- Simple method for incrementally building up basis functions — Just use the value function of the Bellman error

- Rather effective compare to BEBF when $\gamma \to 1$

- Extensions:
  - Deeper hierarchy
  - Multiple secondary learners
  - Incorporating memory for the secondary learner