REPRODUCIBILITY STUDY ON LEARNING FEATURE RELEVANCE THROUGH STEP SIZE ADAPTATION IN TEMPORAL-DIFFERENCE LEARNING

Marwane Adala, Said Ali Kamal Fakhri & Catherine Landry
Department of Computer Engineering
Ecole Polytechnique de Montreal
Montreal, QC 15213, Canada
{marwane.adala,catherine.landry,ali.kamal.fakhri}@polymtl.ca

ABSTRACT

The current paper is a reproducibility study based on the paper: Learning Feature Relevance Through Step Size Adaptation in Temporal-Difference Learning[3]. This paper was not perfectly reproduced, considering some data and parameters issues, such as the β initialization and the exact measure used for the error were not discussed in the paper. However, the general idea of the paper, which is that the TIDBD(λ) (both full-gradient and semi-gradient) and Autostep algorithms are an improvement over regular TD(λ) was still achieved when they converged. These four algorithms were implemented in a GridWorld environment for different step-size values and meta-step sizes. This implementation was successful based on the fact that the state values reached closely resembled the true values.

1 INTRODUCTION AND PROBLEM DEFINITION

The goal of this paper is to conduct a reproducibility study on the paper: Learning feature relevance through step size adaptation in temporal-difference learning by Alex Kearney, Vivek Veeriah, Jadem Travnik, Patrick M. Pilarski and Richard S. Sutton[3]. Although the paper shows experiments for both a GridWorld and a robotic task, only the GridWorld results were reproduced for this study due to lack of access to the data from the robotic task. Four algorithms were implemented to compare their performance against various step-sizes. These algorithms can be found in Appendix A. They are TD, TIDBD with semi-gradient, TIDBD with full-gradient and AutoStep Style Normalized TIDBD. TD has a constant step size, whereas the other listed algorithms dynamically change the step size. AutoStep Style Normalized TIDBD builds on TIDBD by normalizing the meta-weight update and the step-sizes. Although these algorithms include eligibility traces, these were not used for the GridWorld, as the original paper uses $\lambda = 0$ for all implementations. Nonetheless, we described briefly the effect of changing the value of λ in section 4.

This study was done on the GridWorld proposed by Barto in his book [5]. It consists of a simple 5 by 5 grid where the agent can move up, down, left or right. For every transition, the reward is zero, except at state A, where the agent moves to state A' and receive a reward of +10, and state B, where the agent moves to state B' and receive a reward of +5. Transitions that take the agent off the grid leave the state unchanged and result in a reward of -1. The true value of the states are shown on the right part of Figure 1.

Our work is structured into six main sections. We start by the background and motivation in section 2 and the related works in section 3. In section 4, we perform a thorough discussion of the of the Research/Analysis Questions Studied. Finally, we conclude our work in sections 5 and 6 giving some further steps, as well as the link for our code, and we show our contributions in 7.

2 BACKGROUND AND MOTIVATION

The goal of the original paper was to extend IDBD to temporal difference (TD) learning and demonstrate that TD-IDBD (or simply TIDBD) is effective at learning feature relevance. The authors

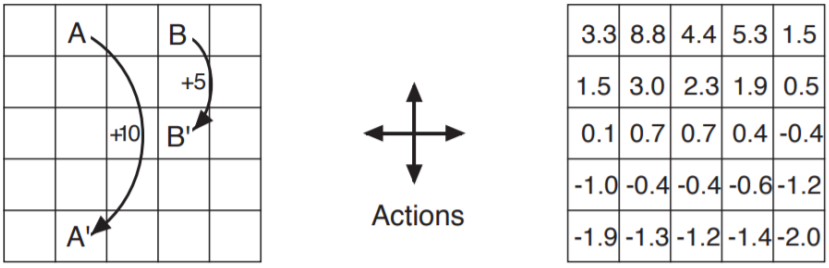

Figure 1: GridWorld

investigated whether TIDBD is better than TD in general, without any vectorization of the step sizes and representation learning. The performance of the methods were evaluated on a 5 by 5 grid-world environment described in Sutton and Barto (1998). The methods were compared for step sizes $\alpha \in (0.0005, 0.5)$ with eligibility traces λ = 0 and discount factor γ = 0.99, and evaluated for meta step-sizes $\theta \in \{0.001, 0.002, ..., 0.02\}$. The semi-gradient TIDBD method outperformed TD for a wide range of θ for all α except at the optimal value (α_0*). In contrast, ordinary gradient TIDBD matched the performance of TD at α_0*. The semi-gradient TIDBD was shown to be less sensitive to θ than the ordinary gradient TIDBD. Both TIDBD methods displayed less sensitivity to θ at higher α_0 values. The ability to adapt α was shown to allow both TIDBD methods to overcome slow convergence when α_0 is low and divergence when α_0 is high. At α_0*, the asymptotic performance of all three methods was relatively the same, although the semi-gradient method converged faster. Next, the authors incorporated the normalization from the AutoStep (an extension of IDBD) to TIDBD to reduce its sensitivity to θ. They extended TIDBD by introducing normalization factors for the updates to the meta weights and the step size, calling the resulting algorithm AutoTIDBD. The performance of AutoTIDBD was then evaluated on the same gridworld environment. AutoTIDBD performed better or as well as TD for all α values. AutoTIDBD also displayed a predictable change in performance for different θ, unlike the previous TIDBD methods. However, the robustness of AutoTIDBD was shown to come at a cost of its learning rate for more aggressive α values.

3 Related works

Three papers were particularly relevant to the completion of this reproducibility study. It is interesting to note that the first two of them were produced by the same university and had in common one author, Alex Kearney, as the reproduced paper.

The first paper [4] presents a comparison of TD and TIDBD. It shows that TIDBD performs better than TD for all values, which was also reported in the original paper studied. This paper had the added value of presenting the variance for both algorithms and it showed that TIDBD greatly helps in reducing the variance. This paper was useful by giving a clear definition of the error used (see subsection 4.3), which is the mean squared value error (MSVE) was calculated at each time-step using the difference between the actual value of all states and the current learned estimate.

$\text{MSVE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$
n = number of steps in an episode
Y_i = estimated state value
$\hat{Y}_i$ = true state value

The second paper [2] once again presents a comparison of TD and TIDBD, but focuses on the application for a robotic arm. Indeed, TIDBD is quite useful for representation learning and feature relevance and scales much better than TD. This paper plotted the accumulated error over all time-steps, instead of the average error. This presented a new view with which to compare the graphs.

The third paper[1] represents a generalization of IDBD for TD methods. This showed that a simple algorithm that uses an online approach to generate an adaptive upper bound on the step-size parameter outperforms previous methods and, unlike all other approaches, totally avoids function

approximation divergence. This confirms that TIDBD outperforms the TD method and is completely consistent with the results of the two previous papers.

All three papers (in addition to the reproduced paper) have in common that they prove that adapting step-sizes during a run leads to better results and a quicker convergence. Indeed, TIDBD always out-performs TD(0), making it a superior algorithm, especially for continuing experiments.

4 Detailed discussion of the Research/Analysis Questions Studied

After reading the paper, the team had many questions, the first one being would it be possible to properly reproduce this experiment? This led to other subsequent questions, which are listed below.

- What is the impact of β for algorithm convergence in TIDBD and AutoTIDBD ?
- For what α values, if any, does TD(λ) outperform AutoTIDBD ?
- How sensitive are semi-gradient and ordinary TIDBD to θ ?
- Is AutoTIDBD more robust than TIDBD with respect to θ ?
- What are AutoTIDBD's downfalls and limitations?
- Does the performance comparison remain consistent for different λ values?

As part of the reproducibility study, four algorithms were implemented on the GridWorld. To ensure the implementation was adequate, the final values obtained for each state after 30,000 steps were plotted for each algorithm. As can be seen in Figure 2, all the methods manage to identify state A and state B as having the maximal value. This step, although not included in the original paper, was found to be a good way to control the sources of errors.

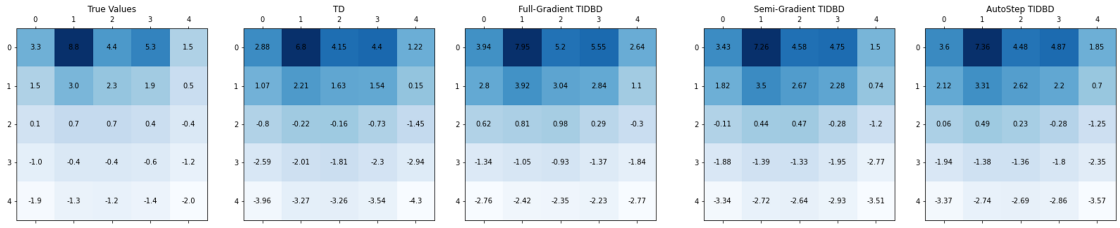

Figure 2: Final estimated values of the algorithms compared to the true value

4.1 Convergence rate comparison for different algorithms

As seen in the plots below, semi-gradient AutoTIDBD and semi-gradient TIDBD are the algorithms that converge the quickest, while TD takes the longest and converges to a higher value. The team was able to reproduce these results, albeit a noisier version. Smoothing seems to have been applied to the original graphs as all values are normalized to one, but no mention was made in the paper regarding this. The implementation done by the team managed to reproduce results in terms of convergence rate for $\alpha = 0.01$ and 0.05, but inconclusive results were reached for $\alpha = 0.5$. It is possible that authors of the original paper initialized a different β for this setting (see section 4.2 for more details).

4.2 Impact of β on convergence

While implementing the algorithms, it was found that the initialization for β had a huge impact on convergence. Although the study mentions to initialize $\beta \in \mathbb{R}^n$ as desired, certain values of β led to divergence, such as $\beta = 0$. An implementation online used $\beta = exp(\alpha)$ and it was found to lead to better convergence. Different β were tested to try to get as close as possible to the figures presented in the paper. Considering the paper did not specify to which β they initialized their algorithms,

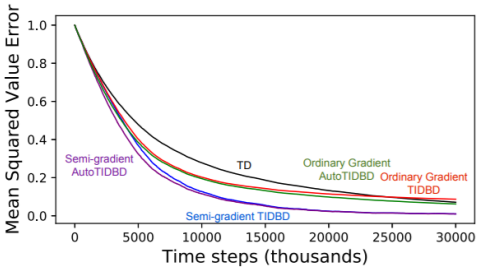

Figure 3: Convergence rate for $\alpha = 0.01$ from the original paper

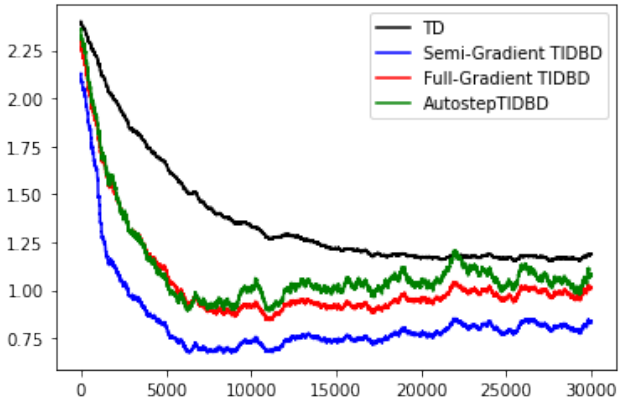

Figure 4: Convergence rate for $\alpha = 0.01$ from our implementation

this can explain the differences in convergence and error of the figures. The final β chosen by the team is $\beta = 1 + log(\alpha)$. Figure 8 and Figure 9 show the importance of initiating β. These graphs compare two β for the same algorithm (AutoTIDBD), with all other hyperparameters equal and the same random seed. One converges somewhat good and the other completely diverges and reaches very high error values.

4.3 Error plotting

Another issue from the paper was plotting the error. As mentionned in section 1, the chosen paper did not specify how the error was calculated or plotted. Another paper [4] confirmed the intuition that the error must be taken between the real value and the current estimation of the value. This was done for each time step. Furthermore, it was necessary to take the error against all state-values, not just the one being updated. The scale on which the error was plotted also seemed to be problematic. Indeed, it is not a linear scale and some values were considered as mistakes (or a typographical error) by the team. For instance, Figures 2, 4 and 5 of the original paper state that they used a step-size of 25e-3, but considering their placement on the scale, it should have been 2.5e-3. The team has made the adjustment in the current paper.

4.4 Sensitivity of semi-gradient and ordinary gradient TIDBD to θ

The results presented by the authors indicate that semi-gradient TIDBD was not sensitive to θ, as opposed to the ordinary gradient TIDBD. This is because the semi-gradient version neglects the effect of changing the weights on the target. Hence, semi-gradient should be more stable by avoiding the non-stationarity in the update of the target value in the weight update.

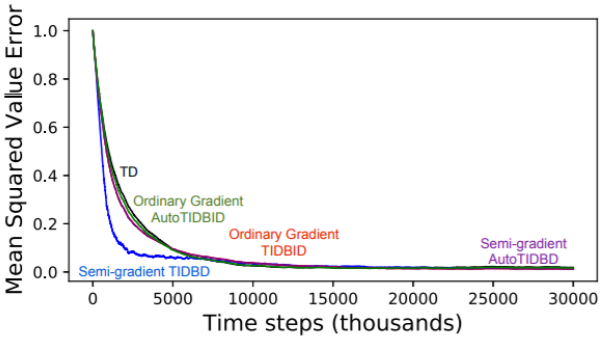

Figure 5: Convergence rate for $\alpha = 0.05$ from the original paper

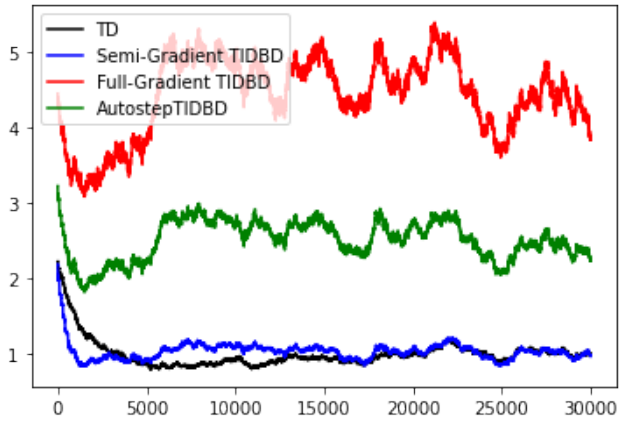

Figure 6: Convergence rate for $\alpha = 0.05$ from our implementation

4.5 Convergence

None of the algorithms examined converged for $\alpha = 0.5$. This can be attributed to how β was initialized by the authors as mentioned in subsection 4.2.

4.6 What are AutoTIDBD's downfalls and limitations?

First and foremost, AutoTIDBD's stability and performance were demonstrated to come at the expense of its learning rate for aggressive α values. AutoTIDBD learns more slowly for aggressive α_0 than TIDBD. Furthermore, AutoTDBD is incompatible with off-policy prediction methods, as well as approaches that use various eligibility traces, including True Online TD.

4.7 Does the performance comparison remain consistent for different λ values?

Our investigation found that changing λ for TD(λ) (algorithm 2) gave faster convergence and reduced the Asymptotic Mean Timestep Error (see Figures 14 and 15). As for algorithm 3 TIDBD(λ) with semi-gradient, it has no effect. For algorithm 4 TIDBD(λ) with full gradient, $\lambda = 1$ make the algorithm invariant to the choice of θ. $\lambda = 0.75$, which is the best configuration let $\alpha = 0.01$ give the least asymptotic mean timestep error. The other values of λ gave similar results (see Figures 16, 17, 18, 19, and 20). Finally for algorithm 6 AutoStep Style Normalized TIDBD(λ) , the more that lambda approaches 1, the more the effect of theta will be reduced (see Figures 21 and 22). These observations confirm, at this level, that the comparison remain consistent for different λ values.

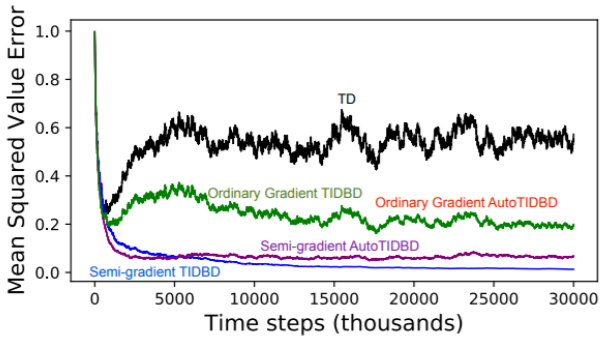

Figure 7: Convergence rate for $\alpha = 0.5$ from the original paper

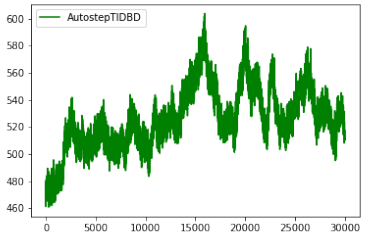

Figure 8: Convergence rate for $\beta_i = 0$

5 Conclusion

The reproducibility study presented above was based on the paper[3]. One of the original paper's main goals was to show that TIDBD (an extension of IDBD to TD) is effective at learning feature relevance. Another goal was to extend TIDBD to AutoTIDBD in order to improve robustness over conventional TD methods with a tuned static step size.

In order to do these two tasks, the team implemented a custom GridWorld environment based on the open AI gym standard and reflecting the one described in Sutton's book**??**. This allowed to seamlessly implement the algorithms and render episodes.

All four algorithms applicable to the GridWorld were implemented. The final learned state-values for TD, TIDBD (full gradient and semi-gradient) and AutoTIDBD were presented at Figure 2 and shown to be very close to the true values.

Plotting the error—i.e., the difference between the learned state value and the true state value, at each time-step reproduced the results shown in the original paper that AutostepTIDBD and TIDBD both converge faster than TD for multiple α values. A contribution from the team was also to clearly define which error the plots used, something that was not explicitly stated in the original paper.

Another significant contribution from the team was showing that β initialization does have an impact on convergence and cannot be initialized arbitrarily. Different values were tried and $\beta = 1 + log(\alpha)$ was chosen to implement all figures seen in the report, unless otherwise specified.

Future contributions would be to pursue this study and plot the affect of meta step-size θ on different algorithms. The original paper had done this study and it would be interesting to see if it is reproducible. Of course, doing this within the robotic arm task would also be a very interesting study, although quite a bit longer.

One other perspective to consider in this context, will be to generalize TIDBD from being based on only one step TD error to a more general multi-step update as in the recent work [7]. We can attempt to change TIDBD so that it can use online RL with eligibility traces to train a deep neural network as a function approximator, as indicated in[7], and therefore investigate multi step-size tuning. A final

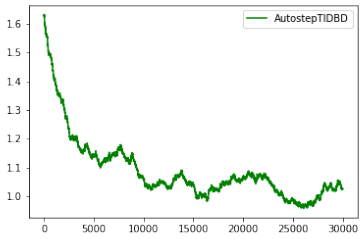

Figure 9: Convergence rate for $\beta_i = 1 + log(\alpha)$

perspective will be also to conduct extensive research on the effect of the parameter λ on TIDBD and AutoTIDBD especially in the robotic arm settings.

6 Link to Code

Everything has been uploaded to our GitHub repository [6], where we can follow the instructions in the README file to reproduce our work.

7 Contributions by each team member

Catherine implemented algorithms for TD(0) and TIDBD (with full-gradient and semi-gradient). She implemented the average error graph and found the right way to calculate the error. As for the report, she wrote the abstract, the introduction and the related works sections. She also wrote part of the discussion and the conclusion.

Ali found an implementation of an existing GridWorld environment and modified it to work with our problem. He also implemented the AutoStep TIDBD and helped troubleshoot various coding issues. He wrote the background and motivation section and some parts of discussion. He also reviewed and edited the report.

Marwane helped both Catherine and Ali as needed. Moreover, he dealt with the part concerning reproducing the work. He also contributed in the introduction, the Detailed discussion of the Research/Analysis Questions, in the related works part and in the conclusion. For coding, he proposed a first version for TIDBD(λ) with semi-gradient based on Catherine's work. Finally, he reviewed the report, managed the references and the appendix, and tried to debug the code for the differences seen in the results between the original work and the reproduced one.

We hereby state that all the work presented in this report is that of the authors.

References

[1] William Dabney and Andrew G Barto. "Adaptive step-size for online temporal difference learning". In: *Twenty-Sixth AAAI Conference on Artificial Intelligence*. 2012.

[2] Johannes Günther et al. "Examining the use of temporal-difference incremental delta-bar-delta for real-world predictive knowledge architectures". In: *Frontiers in Robotics and AI* 7 (2020), p. 34.

[3] Alex Kearney et al. "Learning feature relevance through step size adaptation in temporal-difference learning". In: *arXiv preprint arXiv:1903.03252* (2019).

[4] Alex Kearney et al. "Tidbd: Adapting temporal-difference step-sizes through stochastic meta-descent". In: *arXiv preprint arXiv:1804.03334* (2018).

[5] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[6] The Project Team. *Github Repo for our Project*. 2021. URL: `https://github.com/marsup13/RL_project_INF8953DE`.

[7] Kenny Young, Baoxiang Wang, and Matthew E Taylor. “Metatrace actor-critic: Online step-size tuning by meta-gradient descent for reinforcement learning control”. In: *arXiv preprint arXiv:1805.04514* (2018).

A APPENDIX A

Algorithm 2 TD(λ)

Initialize vectors $z \in 0^n$, and both $w \in \mathbb{R}^n$; initialize a small scalar $\alpha > 0$; observe state S
Repeat for each observation s' and reward R:
  $\delta \leftarrow R + \gamma w^\top \phi(s') - w^\top \phi(s)$
  For $i = 1, 2, \cdots, n$:
    $z_i \leftarrow z_i \gamma \lambda + \phi_i(s)$
    $w_i \leftarrow w_i + \alpha_i \delta z_i$
    $s \leftarrow s'$

Figure 10: Algorithm 2: TD(λ)

Algorithm 3 TIDBD(λ) with semi-gradient

Initialize vectors $h \in 0^n$, $z \in 0^n$, and both $w \in \mathbb{R}^n$ and $\beta \in \mathbb{R}^n$ as desired; initialize a scalar θ; observe state S
Repeat for each observation s' and reward R:
  $\delta \leftarrow R + \gamma w^\top \phi(s') - w^\top \phi(s)$
  For element $i = 1, 2, \cdots, n$:
    $\beta_i \leftarrow \beta_i + \theta \delta \phi_i(s) h_i$
    $\alpha_i \leftarrow e^{\beta_i}$
    $z_i \leftarrow z_i \gamma \lambda + \phi_i(s)$
    $w_i \leftarrow w_i + \alpha_i \delta z_i$
    $h_i \leftarrow h_i [1 - \alpha_i \phi_i(s) z_i]^+ + \alpha_i \delta z_i$
  $s \leftarrow s'$

Figure 11: Algorithm 3: TIDBD(λ) with semi-gradient

Algorithm 4 TIDBD(λ)

Initialize vectors $h \in 0^n$, $z \in 0^n$, and both $w \in \mathbb{R}^n$ and $\beta \in \mathbb{R}^n$ as desired; initialize a scalar θ; observe state S
Repeat for each observation s' and reward R:
  $\delta \leftarrow R + \gamma w^\top \phi(s') - w^\top \phi(s)$
  For element $i = 1, 2, \cdots, n$:
    $\beta_i \leftarrow \beta_i - \theta \delta [\gamma \phi(s') - \phi(s)] h_i$
    $\alpha_i \leftarrow e^{\beta_i}$
    $z_i \leftarrow z_i \gamma \lambda + \phi_i(s)$
    $w_i \leftarrow w_i + \alpha_i \delta z_i$
    $h_i \leftarrow h_i [1 + \alpha_i z_i [\gamma \phi_i(s') - \phi_i(s)]]^+ + \alpha_i \delta z_i$
  $s \leftarrow s'$

Figure 12: Algorithm 4: TIDBD(λ) with full gradient

Algorithm 6 AutoStep Style Normalized TIDBD(λ)

Initialize vectors $h \in 0^n$, $z \in 0^n$, and both $w \in \mathbb{R}^n$ and $\beta \in \mathbb{R}^n$ as desired; initialize a scalar θ; observe state S
Repeat for each observation s' and reward R:
  $\delta \leftarrow R + \gamma w^\top \phi(s') - w^\top \phi(s)$
  For element $i = 1, 2, \cdots, n$:
    $\eta_i \leftarrow \max[$
      $|\delta[\gamma\phi_i(s') - \phi_i(s)]h_i|,$
      $\eta_i - \frac{1}{\tau}\alpha_i[\gamma\phi_i(s') - \phi_i(s)]z_i(|\delta\phi_i(s)h_i| - \eta_i)]$
  For element $i = 1, 2, \cdots, n$:
    $\beta_i \leftarrow \beta_i - \theta\frac{1}{\eta_i}\delta[\gamma\phi_i(s')) - \phi_i(s)]h_i$
    $M \leftarrow max(-e^{\beta_i}[\gamma\phi_i(s') - \phi_i(s)]^\top z_i,\ 1)$
    $\beta_i \leftarrow \beta_i - \log(M)$
    $\alpha_i \leftarrow e^{\beta_i}$
    $z_i \leftarrow z_i\gamma\lambda + \phi_i(s)$
    $w_i \leftarrow w_i + \alpha_i\delta z_i$
    $h_i \leftarrow h_i[1 + \alpha_i[\gamma\phi_i(s') - \phi_i(s)]z_i]^+ + \alpha_i\delta z_i$
  $s \leftarrow s'$

Figure 13: Algorithm 6: AutoStep style Normalized TIDBD(λ)

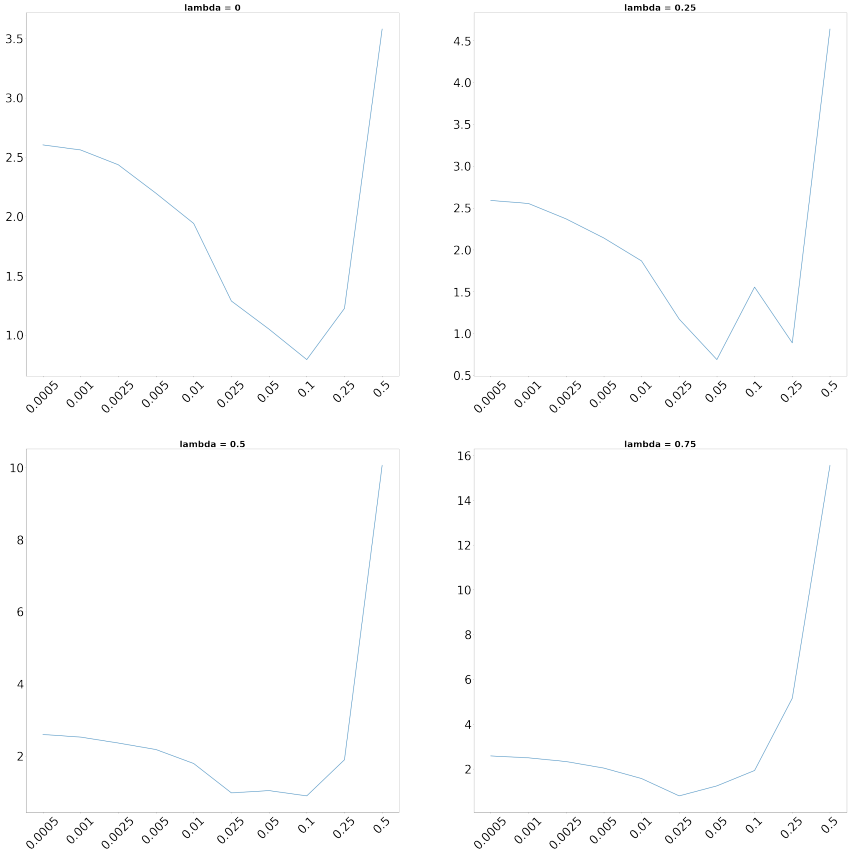

Figure 14: Asymptotic Mean Timestep Error as a function of alpha, with different values of Lambdas in TD(λ)

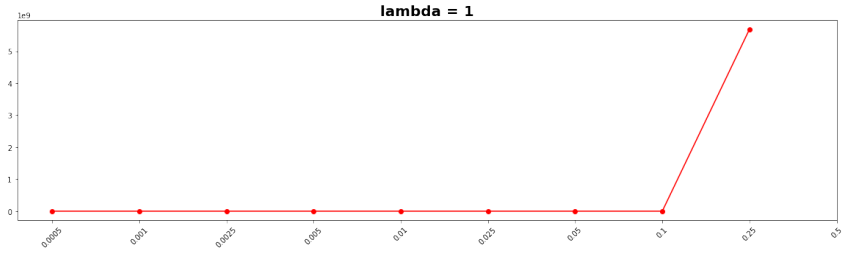

Figure 15: Asymptotic Mean Timestep Error as a function of alpha, with Lambda =1 in TD(λ)

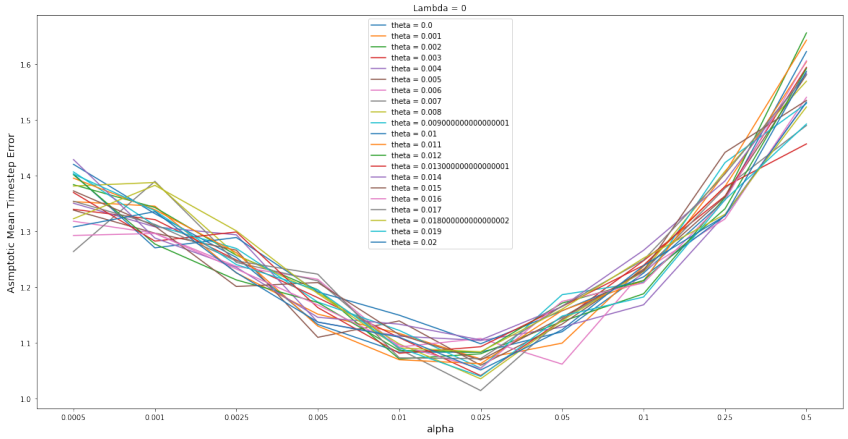

Figure 16: Asymptotic Mean Timestep Error as a function of alpha, Lambda =0 in TIDBD(λ)

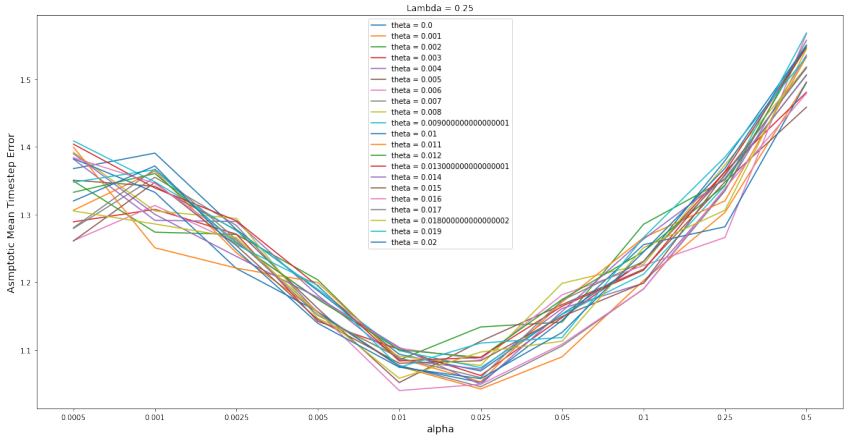

Figure 17: Asymptotic Mean Timestep Error as a function of alpha, with Lambda =0.25 in TIDBD(λ)

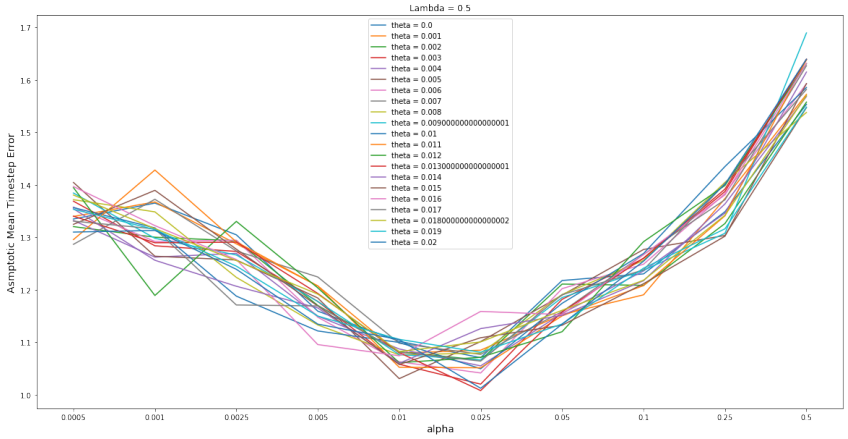

Figure 18: Asymptotic Mean Timestep Error as a function of alpha, with Lambda = 0.5 in TIDBD(λ)

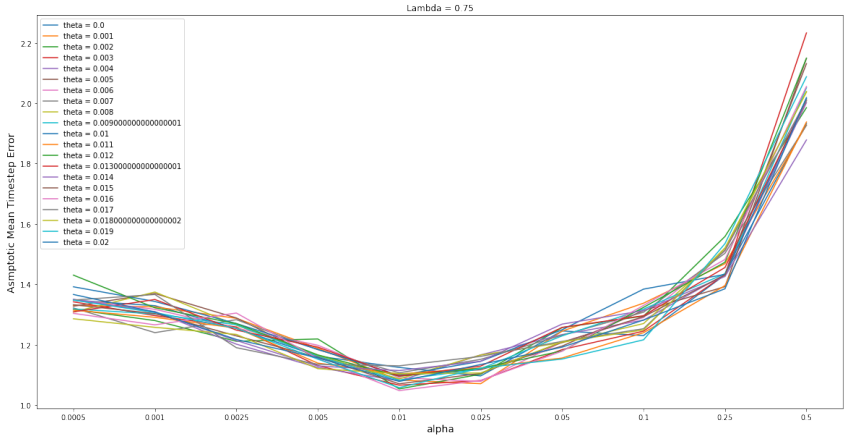

Figure 19: Asymptotic Mean Timestep Error as a function of alpha, with Lambda =0.75 in TIDBD(λ)

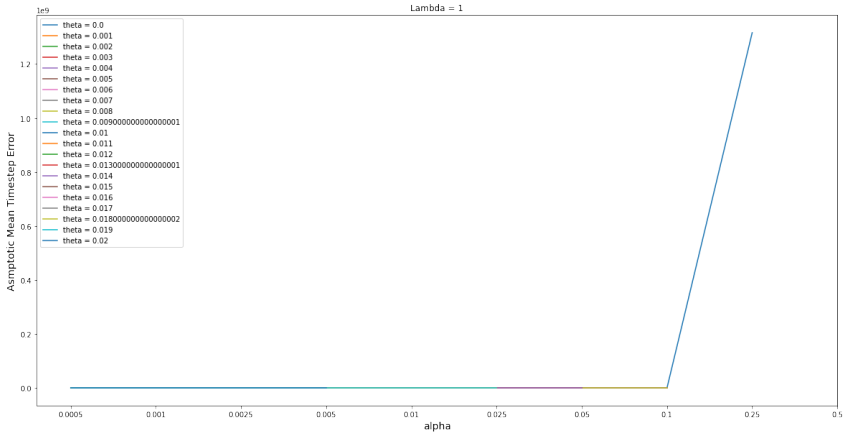

Figure 20: Asymptotic Mean Timestep Error as a function of alpha, with Lambda =1 in TIDBD(λ)

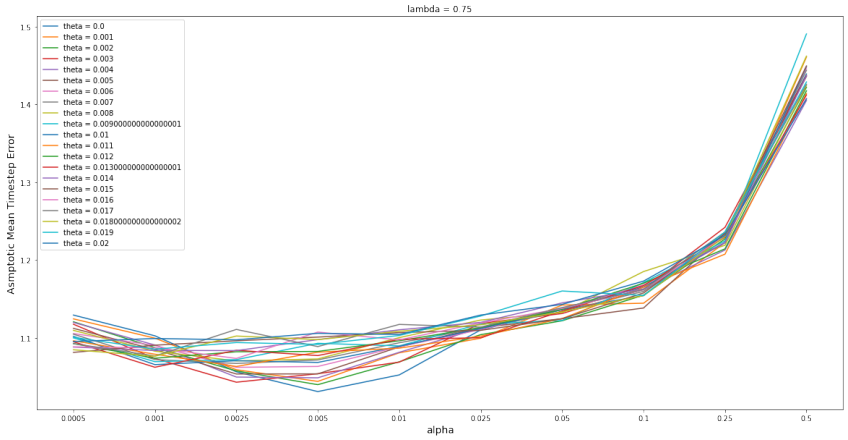

Figure 21: Asymptotic Mean Timestep Error as a function of alpha, with Lambda =0.75 in AutoStep Style Normalized TIDBD(λ)

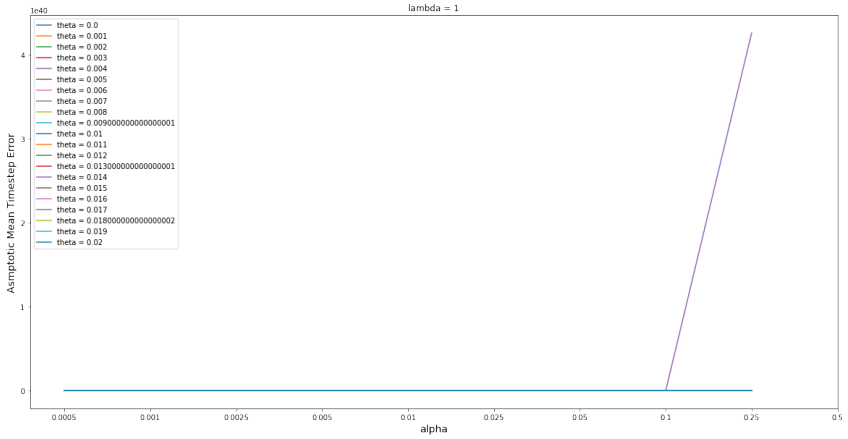

Figure 22: Asymptotic Mean Timestep Error as a function of alpha, with Lambda =1 in AutoStep Style Normalized TIDBD(λ)