Proofs of Some Conclusions in the Paper

May 1, 2023

1 Proof of Backward Testing Oracle

Given the seed initial state s_0^{seed} and its final state s_T^{seed} from running forward simulation $f : s_T = f(s_0)$, we first add a slight mutation amount Δs towards s_0^{seed} and obtain $s_0^{mut} = s_0^{seed} + \Delta s$. We define the optimization objective function $h(\Delta s)$ as the Euler distance between the seed final state $f(s_0^{seed})$ and the mutant's final state $f(s_0^{seed} + \Delta s)$:

$$h(\Delta s) = \|f(s_0^{seed}) - f(s_0^{seed} + \Delta s)\| \tag{1}$$

Since Δs is small and the forward function f is differentiable (an assumption listed in our paper), function h can be approximated by a second-order Tylor Expansion:

$$h(\Delta s) \approx h(0) + c^T \Delta s + \frac{1}{2}\Delta s^T \Lambda \Delta s \tag{2}$$

, where Λ is an $n \times n$ symmetric Hessian Matrix of h. Since $h(0) = \|f(s_0^{seed}) - f(s_0^{seed} + 0)\| = 0$, Eq. (2) can be further writen as:

$$h(\Delta s) \approx c^T \Delta s + \frac{1}{2}\Delta s^T \Lambda \Delta s \tag{3}$$

Denote the gradient of function h w.r.t. Δs as g, we run gradient-descend algorithm on Δs:

$$\Delta s_{k+1} = \Delta s_k - \alpha_k g_k \tag{4}$$

, where k is the iteration number of gradient-descend, and s_k, g_k is the mutation amount and gradient of h at iteration k, respectively. Since $g_k = \Lambda \Delta s_k + c$ and the function h is (approximately) quadratic, the α_k that minimizes $h(\Delta s_{k+1})$ is explicitly given by:

$$\alpha_k = \frac{g_k^T g_k}{g_k^T \Lambda g_k} \tag{5}$$

With the α_k given above, we can obtain the decrease amount of $h(\Delta s)$ between iteration k and $k+1$ as:

$$h(\Delta s_{k+1}) - h(\Delta s_k) = -\frac{(g_k^T g_k)^2}{2 g_k^T \Lambda g_k} \tag{6}$$

Denote the optimal point of function h as Δs^*, we have the following property considering that h is approximately quadratic:

$$h(\Delta s) - h(\Delta s^*) = \frac{1}{2} g(\Delta s)^T \Lambda^{-1} g(\Delta s) \tag{7}$$

Plugging in Eq. (7), we can derive the change rate of the decrease amount between nearby iterations:

$$\frac{[h(\Delta s_{k+1}) - h(\Delta s^*)] - [h(\Delta s_k) - h(\Delta s^*)]}{h(\Delta s_k) - h(\Delta s^*)} = -\frac{(g_k^T g_k)^2}{(g_k^T \Lambda g_k)(g_k^T \Lambda^{-1} g_k)} \tag{8}$$

Hence,

$$h(\Delta s_{k+1}) - h(\Delta s^*) = \left[1 - \frac{(g_k^T g_k)^2}{(g_k^T \Lambda g_k)(g_k^T \Lambda^{-1} g_k)}\right] [h(\Delta s_k) - h(\Delta s^*)] \tag{9}$$

Next, we utilize the Kantorovich inequality, which says that for all $x \in \mathbb{R}^n \backslash \{0\}$ and eigenvalues $0 < \lambda_n \leq \cdots \leq \lambda_1$:

$$\frac{(x^T x)^2}{(x^T A x)(x^T A^{-1} x)} \geq \frac{4\lambda_1 \lambda_n}{(\lambda_1 + \lambda_n)^2} \tag{10}$$

Thus we have:

$$h(\Delta s_{k+1}) - h(\Delta s^*) \leq \left[1 - \frac{4\lambda_1 \lambda_n}{(\lambda_1 + \lambda_n)^2}\right] [h(\Delta s_k) - h(\Delta s^*)] \tag{11}$$

Eq. (11) can be further simplified to:

$$h(\Delta s_{k+1}) - h(\Delta s^*) \leq \frac{(\lambda_1 - \lambda_n)^2}{(\lambda_1 + \lambda_n)^2} [h(\Delta s_k) - h(\Delta s^*)] \tag{12}$$

Denote $\kappa = \lambda_n / \lambda_1$, we arrive at:

$$h(\Delta s_{k+1}) - h(\Delta s^*) \leq \frac{(\kappa - 1)^2}{(\kappa + 1)^2} [h(\Delta s_k) - h(\Delta s^*)] \tag{13}$$

Denote $\gamma = (\kappa - 1)^2 / (\kappa + 1)^2$. Since $0 < \lambda_n \leq \lambda_1$, we can get:

$$\gamma = \frac{(\kappa - 1)^2}{(\kappa + 1)^2} \in [0, 1) \tag{14}$$

Plugging in Eq. (14), we can see that:

$$\lim_{k \to \infty} h(\Delta s_k) = \lim_{k \to \infty} \gamma^k h(\Delta s_0) = 0 \tag{15}$$

2 Proof of Theorem 2

Proof of Theorem 2. Refer to the number of times that the time stepping function TS is applied as k, we have:

1. For $k = 0$, the state s_k is s_{meta}, which is a valid state by assumption.

2. Assuming for $k = n-1$, state s_k is a valid state. By the definition of time-stepping function TS, $s_{k+1} = TS(s_k, \tau_k)$. Under the assumption that the function TS conforms to the physical laws (one of the assumptions of Theorem 2; also see the discussion of the paper for situations when TS is buggy), and s_k is a valid state, the derived state s_{k+1} should also be a valid state.

□