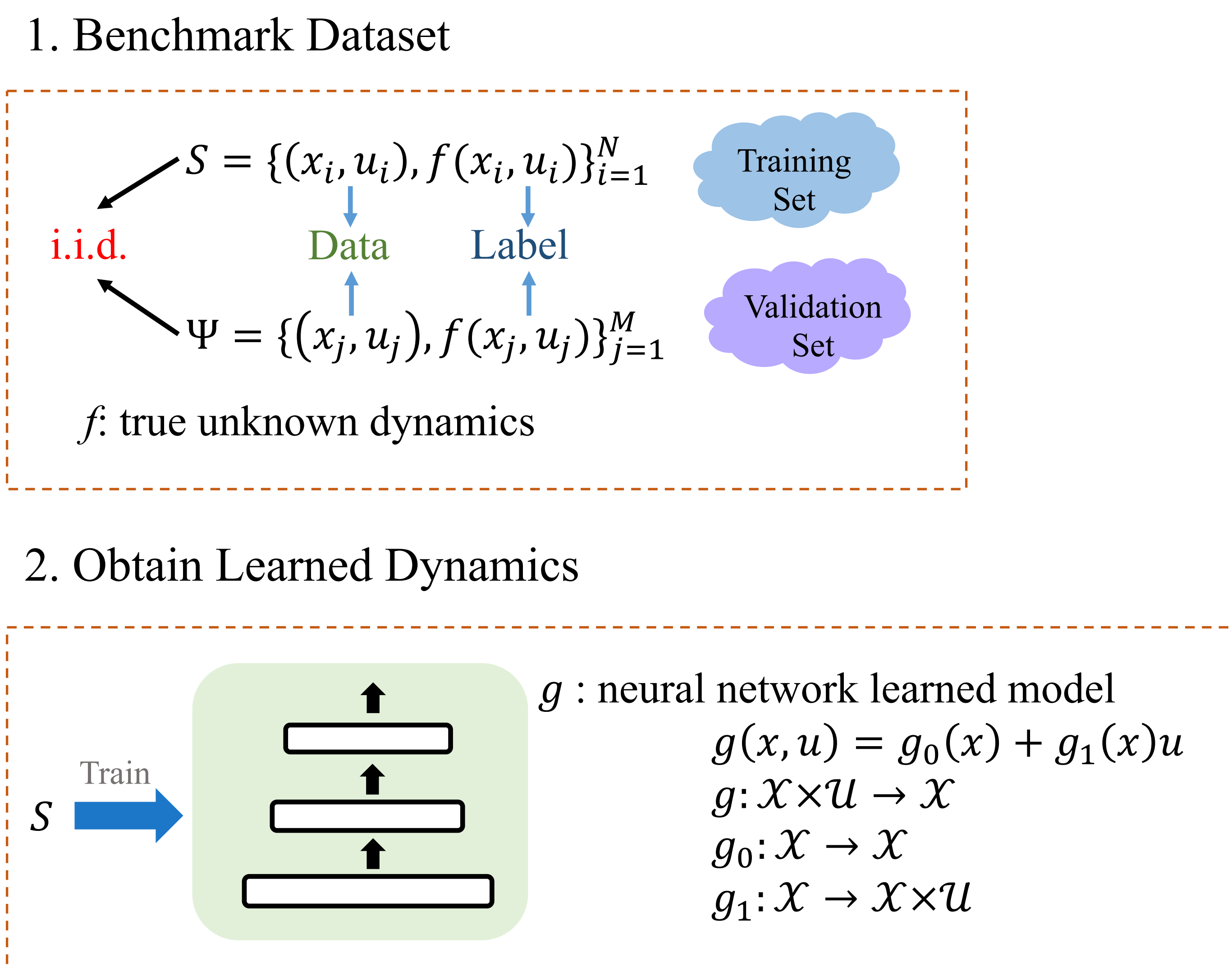
LMTD-RRT Learning Poster

(Reference: Knuth, Craig, et al. "Planning with learned dynamics: Probabilistic guarantees on safety and reachability via lipschitz constants." IEEE Robotics and Automation Letters 6.3 (2021): 5129-5136.)


1. Benchmark Dataset
$S = \{(x_i, u_i), f(x_i, u_i)\}_{i=1}^{N}$
Training Set
i.i.d.
Data
Label
$\Psi = \{(x_j, u_j), f(x_j, u_j)\}_{j=1}^{M}$
Validation Set
f: true unknown dynamics
2. Obtain Learned Dynamics
S
Train
g : neural network learned model
$g(x, u) = g_0(x) + g_1(x)u$
$g: \mathcal{X} \times \mathcal{U} \to \mathcal{X}$
$g_0: \mathcal{X} \to \mathcal{X}$
$g_1: \mathcal{X} \to \mathcal{X} \times \mathcal{U}$

3. Bound Model Error in Trusted Domain

Initialization:

$$S_D = \{(\bar{x},\bar{u}) \in S \,|\, \|f(\bar{x},\bar{u}) - g(\bar{x},\bar{u})\| \le \mu + a\sigma\}$$
$$r = \mu + a\sigma$$

Loop:

$$D = \bigcup_{(\bar{x},\bar{u})\subset S_D} B_r(\bar{x},\bar{u})$$

Estimate Lipschitz Constant

$D \cap \Psi \xrightarrow{\text{Divide}}$ N_s pairwise sample sets, each set has N_L pairs of i.i.d.:

$$\{z_j\}_{j=1}^{N_s} = \{z_{1,j}, \dots, z_{i,j}, \dots, z_{N_L,j}\}_{j=1}^{N_s}, \quad z_{i,j} = \left\{z_1^{i,j}, z_2^{i,j}\right\} = \{(x,u)_1^{i,j}, (x,u)_2^{i,j}\}$$

$\{z_j\}_{j=1}^{N_s}$ ->Lipschitz Constants-> $\{L_{f-g}^{j}\}_{j=1}^{N_s}$ ->L Converge to Reverse Weibull-> $\{\gamma_j\}_{j=1}^{N_s}$

1st Upper Bound

2nd Upper Bound

->Maximum Likehood to N_s Sets->$\hat{\gamma}, \xi$

if KS-test($\hat{\gamma}$):

$$L_{f-g} = \hat{L}_{f-g} = \hat{\gamma} + \boxed{\Phi^{-1}(\rho)^{\xi}}$$

Overestimate with Probability ρ

Overestimate $\hat{L}_{f-g}$ → Somehow Conservative

Determine Trusted Domain

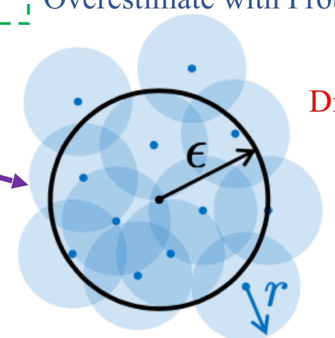
$\boxed{L_{f-g} \ge 1 \rightarrow fail}$

$\epsilon = L_{f-g} b_T + e_T$

$\boxed{\text{if } r > \epsilon\text{: return } r \text{ and } D}$

else: $r \leftarrow \epsilon + \alpha$

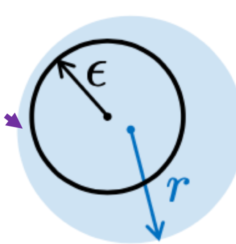
Use $D \cap \Psi$ to obtain L_{g_0} and L_{g_1}



Difficult to check when planning:

$$\bigcup_{(\bar{x},\bar{u})\in\mathcal{W}} \mathcal{B}_r(\bar{x},\bar{u}) \supset \mathcal{B}_\epsilon(x,u)$$

Hard not check planned points within D_ϵ



Easy to check planned points within D_ϵ:

(if $r > \frac{e_T}{1-L_{f-g}}$)

1. Find the nearest neighbor
2. Check $\|(x,u) - (\bar{x},\bar{u})\| \le r - \epsilon$

4. Motion Planning with One Step Feedback Law

Build RRT Tree in Loop (Then construct path after building the tree):

1. Sample new state x_{new}
2. Check staying inside trusted domain:
   a. Find the nearest neighbor $x_{neighbor}$ in S_x
   b. Check distance between x_{new} and $x_{neighbor}$ no more than $r - \epsilon$
   c. Find nearest neighbor $x = x_{near}$ of x_{new} in RRT tree $\mathcal{T}$
   d. For each $\tilde{u}$, find the "NN" $(x_{neighbor}, \tilde{u})$ in S_D, check distance between $(x_{near}, \tilde{u})$ and $(x_{neighbor}, \tilde{u})$ no more than $r - \epsilon$
   e. For each x_{next} and it's "NN" neighbor in S_x, check distance $\leq r - \epsilon$
3. Next state prediction in $N_{samples}$ times loop:
   a. Sample $\tilde{u}$
   b. Check current state-action pair staying inside trusted domain
   c. Check one step feedback law:
      c.1 Perturbed linear equation:
      $x_{next} = g(\tilde{x}, \tilde{u}) = (g_0(x) + \Delta_0) + (g_1(x) + \Delta_1)\tilde{u}$, $\|\Delta_0\| \leq L_{g_0}\epsilon$, $\|\Delta_1\| \leq L_{g_1}\epsilon$
      c.2 $A = g_1(x) + \Delta_1$, $b = x_{next} - (g_0(x) + \Delta_0)$
      -> Constrained least squares: $\min_{\tilde{u} \in u} \|A\tilde{u} - b\|_2^2$
      c.3 Bound $\tilde{u}$ to satisfy $\|u - \tilde{u}\| \leq \frac{\|g_1(x)^+\|(\|\Delta_1\|\|u\| + \|\Delta_0\|)}{1 - \|g_1(x)^+\|\|\Delta_1\|}$
   d. Check next state staying inside trusted domain
   e. Check x_{next} within X_{safe}
   f. Check newly predicted x_{next} closer to x_G, replace old prediction if true
   g. $\mathcal{T} \leftarrow \mathcal{T} \cup \{x_{best}\}$

ensuring safety and invariance about the goal

5. Supplements and Personal Thinking

Supplement: Proof of perturbed linear equation:

1. Since $\|g_0(\tilde{x}) - g_0(x)\| \le L_{g_0}\|\tilde{x} - x\|$: $g_0(\tilde{x}) \le g_0(x) + L_{g_0}\|\tilde{x} - x\|$
2. Since $x = f(x', u'), \tilde{x} = f(\tilde{x}', \tilde{u}')$ an $\forall (x,u) \in D \quad f(x,u) = g(x,u) + \delta, \quad \|\delta\| \le \epsilon$
$$g_0(\tilde{x}) \le g_0(x) + L_{g_0}\epsilon$$
3. Same to g_1, then:
$$g(\tilde{x}, \tilde{u}) \le (g_0(x) + L_{g_0}\epsilon) + (g_1(x) + L_{g_1}\epsilon)\tilde{u}$$

Personal thinking about shortages and future work:

1. Noise and real-world sys -> Lipschitz constants big -> radius of B_ϵ shrink -> too conservative:
    a. Actually twice upper bound and overestimate confidence lead to somehow conservative.
    b. And planning limit in D_ϵ can lead to further more conservative.
    c. Fortunately, let $L_{f-g} < 1$ can ease this conservatism.
    d. Maybe noise sampling and randomness modeling will be needed.
2. This work under hypothesis that the true dynamics are deterministic. For stochastic dynamics, estimating the Lipschitz constant of mean dynamics and appropriate noise modeling are necessary (mentioned in paper).
3. More complicated modeling for extending to underactuated sys (mentioned in paper).
4. Normalization can be more considered (mentioned at MIT seminar).
5. Use Lipschitz constants estimation for estimating uncertainty of ensembles (learning and planning algorithms ensembles), so that better bridging and motion planning (at MIT seminar).
6. Much hypothesis (need prune), it is strictly restricted and hard for rue sys application.