Learning Step-Size Adaptation in CMA-ES

Gresa Shala, André Biedenkapp, Noor Awad, Steven Adriaensen,
Marius Lindauer, and Frank Hutter

In a Nutshell

- **Step-size** in CMA-ES must be adapted dynamically
- Using **Guided Policy Search (GPS)** learn to control step-size offline in an automated, data-driven way
- Learned policies **generalize** beyond training setting
  - **higher dimensions**
  - **longer runs**
  - **other function classes**

Related Work

- **Algorithm Configuration**
  - Static [e.g. Ansótegui et al. 2009, Hutter et al. 2011, López-Ibáñez et al. 2011]
  - Dynamic [e.g. Adriaensen et al. 2016, Biedenkapp et. al 2020]
- **Parameter Control Using Reinforcement Learning (RL)**
  - Online [Muller et al. 2002, Pettinger and Everson 2002, Chen et al. 2005, Eiben et al. 2007, Sakurai et al. 2010, Gaspero and Urli 2012, Karafotias, et al. 2014]
  - Offline [Battiti and Campigotto 2012, Sharma et al. 2019]
- **Learning to Optimize** [Li and Malik 2017]

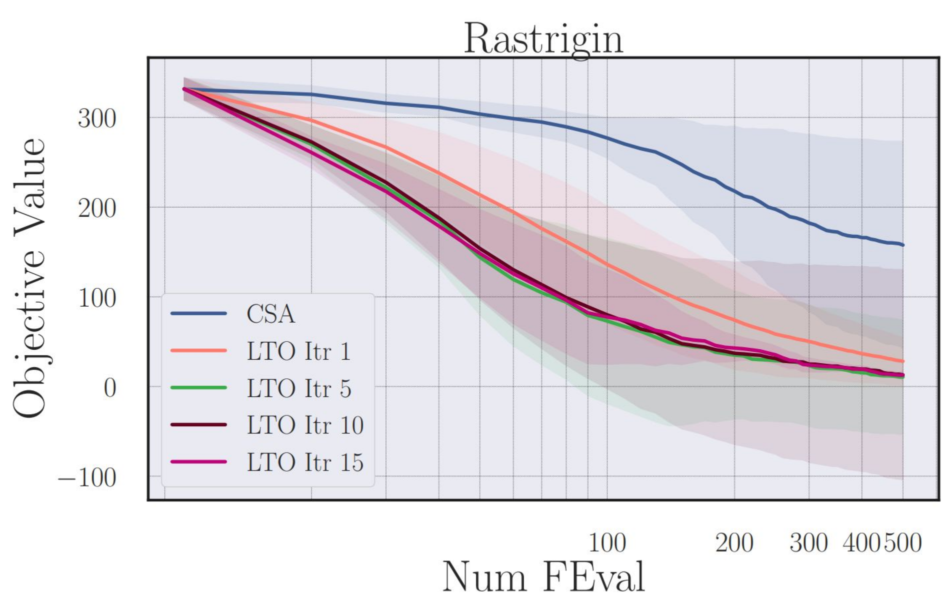
GPS for DAC

Dynamic Algorithm Configuration (DAC)

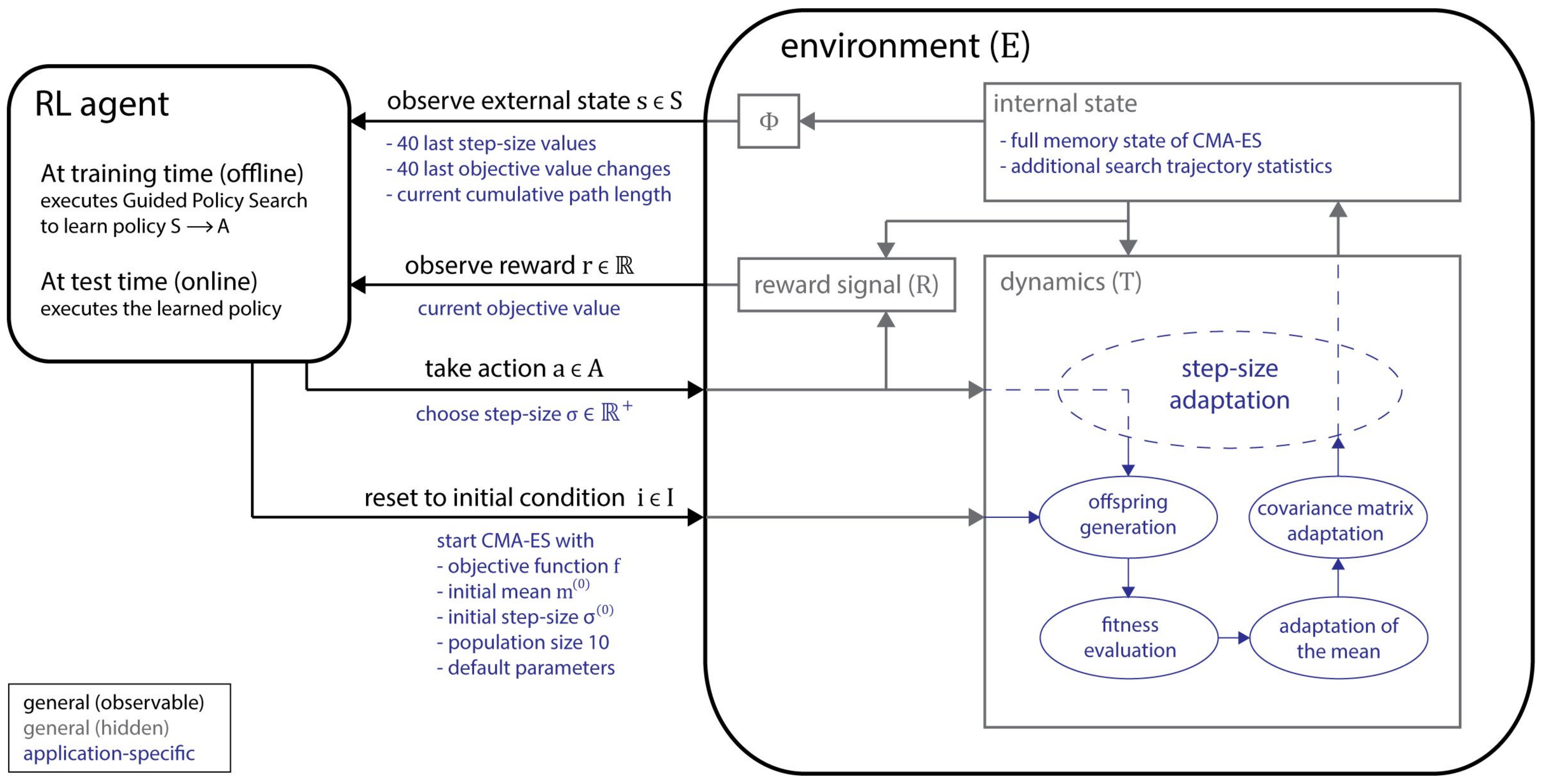
- Configure **per time-step & per-instance**
- Learn a configuration policy
- Can be posed as **RL problem**
- Prior-art: Value-based RL (DQN)
  - Not sample-efficient
  - Focus on categorical parameters
  - Learning from scratch

Guided Policy Search (GPS)

- **Sample-efficient** RL method from robotics
- Learn arbitrary parameterized policies
- Represent policies as neural networks
- Learn policies offline
- Easily warm-started from demonstration
- Combines
  - **Imitation learning** (supervised ML)
  - **Learning from a reward signal** (RL)

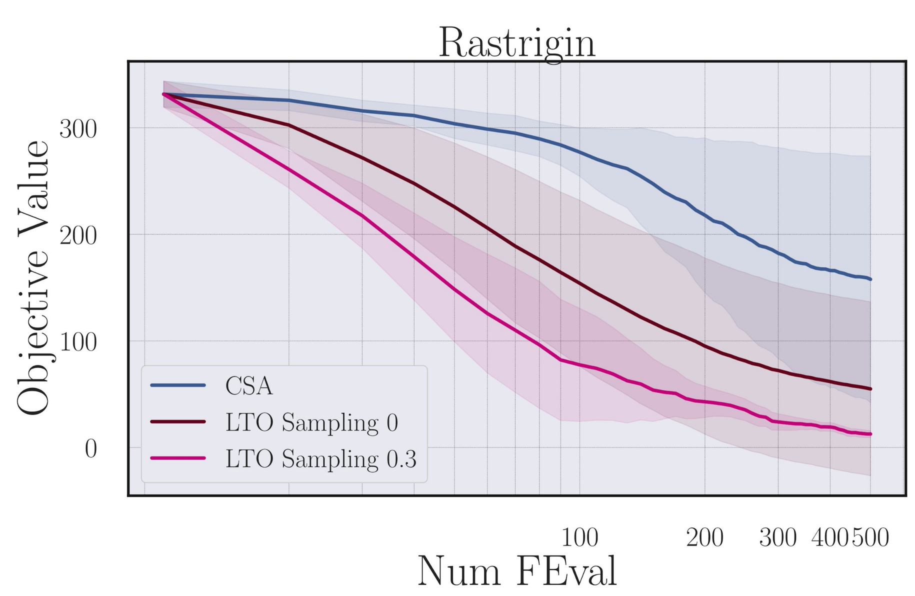


Learning Step-Size Adaptation



Learning from a Hand-Crafted Heuristic

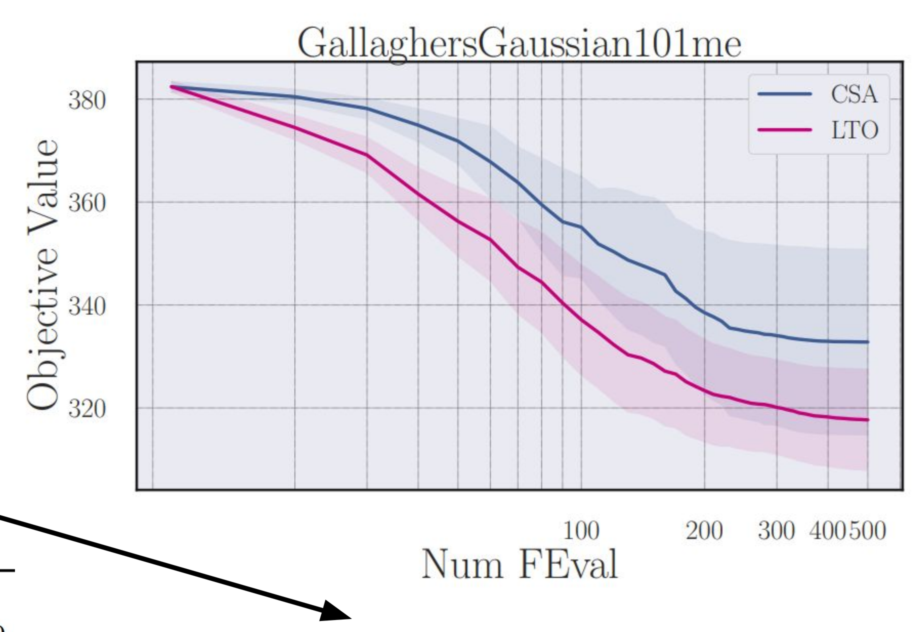
- Learn from **Cumulative Step-size Adaptation** (CSA)
- Vanilla GPS uses example trajectories **only once**, in the beginning, to warm-start the search.
- We **repeatedly query** the hand-crafted baseline
  - Continuous use of expert knowledge
  - Learn from the teacher in many more situations
  - Sampling rate:
    0.0 ⟶ Vanilla GPS
    1.0 ⟶ Pure imitation learning
    0.3 ⟶ Good trade-off

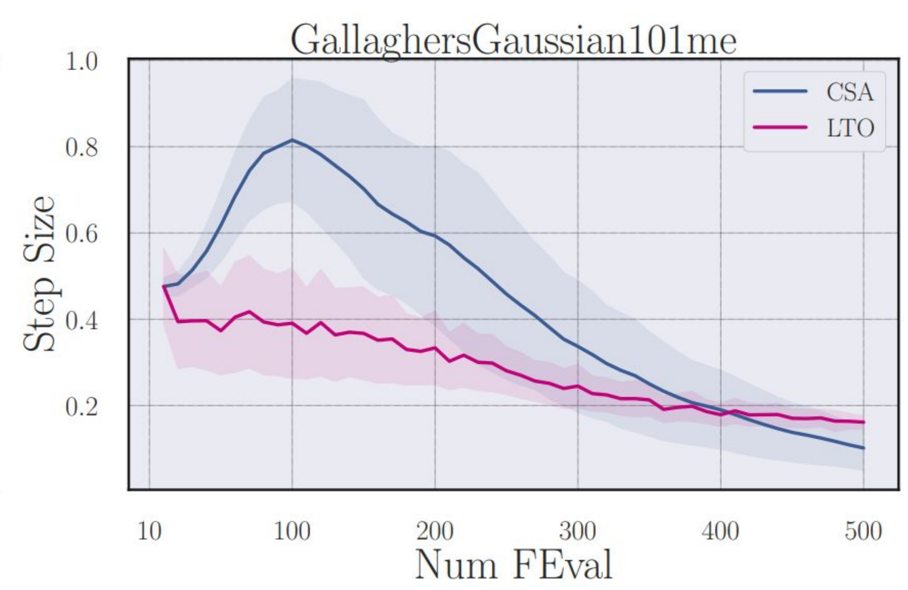


Performance & Generalization

The learned policies ...

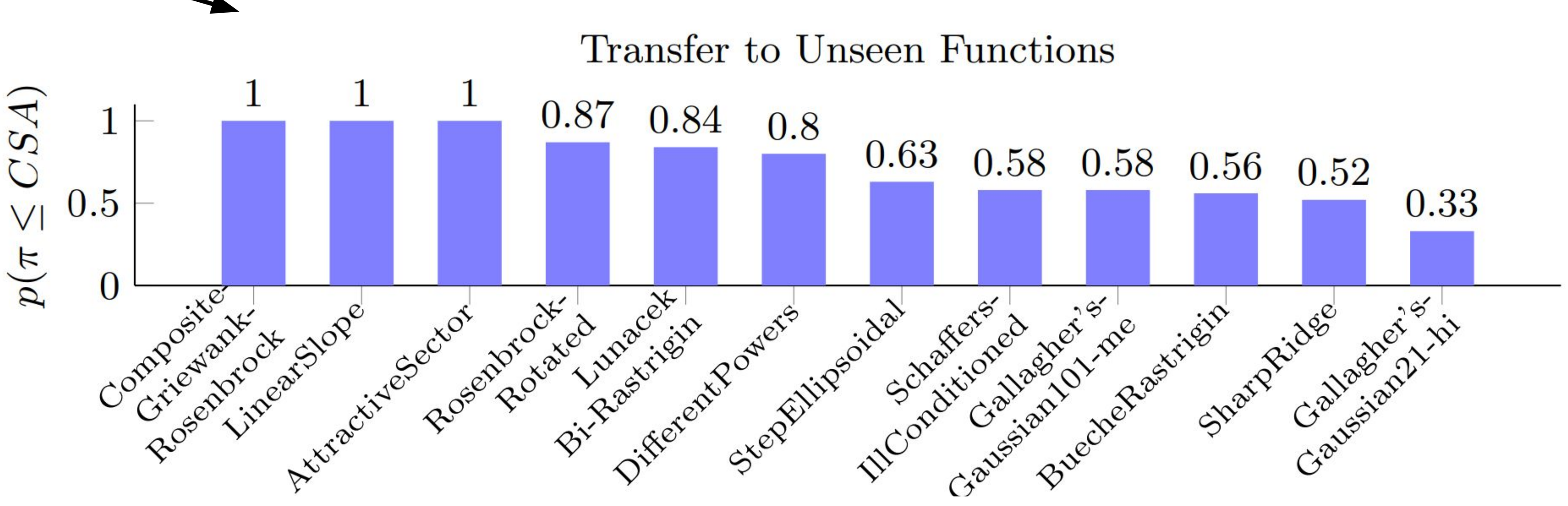
- are capable of producing **well-performing** step-sizes ⟶
- **generalize** to
  - longer trajectories
  - higher dimensions
  - other function classes





	Trajectory Length							Dimensions					
	50	100	150	200	250	500	1000	35	40	45	50	55	60
BentCigar	**0.89**	0.00	0.00	0.00	0.00	0.05	0.04	0.87	0.98	0.56	0.49	0.76	**1.00**
Discus	0.90	**0.95**	0.76	0.40	0.00	0.00	0.00	0.89	0.86	0.93	0.94	0.94	**0.97**
Ellipsoid	**0.94**	0.92	0.90	0.86	0.61	0.00	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Katsuura	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.92	0.92	0.96	**1.00**	0.96	0.87
Rastrigin	**1.00**	0.81	0.80	0.83	0.92	0.73	0.74	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Rosenbrock	**0.93**	0.77	0.78	0.90	0.62	0.24	0.04	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Schaffers	**0.60**	0.55	0.40	0.39	0.48	0.39	0.57	0.31	0.58	0.78	**0.87**	0.76	0.74
Schwefel	**0.99**	0.52	0.76	0.79	0.87	0.84	0.65	**1.00**	0.96	0.96	**1.00**	**1.00**	0.98
Sphere	**0.89**	0.00	0.00	0.00	0.00	0.00	0.00	0.41	0.38	0.56	0.65	0.64	**0.72**
Weierstrass	0.97	0.97	0.89	0.92	**1.00**	**1.00**	**1.00**	0.97	**1.00**	0.95	**1.00**	**1.00**	0.93
Average	0.91	0.65	0.63	0.61	0.55	0.43	0.40	0.84	0.87	0.87	0.89	0.91	0.92

(a) Different Trajectory Lengths (b) Different # Dimensions



@automlfreiburg