plot_mean_score

October 7, 2019

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm, colors, rcParams

import bayesmark.constants as cc
import bayesmark.xr_util as xru
from bayesmark.serialize import XRSerializer
from bayesmark.constants import ITER, METHOD, ARG_DELIM
from bayesmark.path_util import abspath
from bayesmark.util import preimage_func
```

[2]:
```
# User settings, must specify location of the data to make plots here for this␣
 ↪to run
DB_ROOT = abspath(".")
DBID = "bo_example_folder"
```

[3]:
```
# Matplotlib setup
# Note this will put type-3 font BS in the pdfs, if it matters
rcParams["mathtext.fontset"] = "stix"
rcParams["font.family"] = "STIXGeneral"
```

[4]:
```
def build_color_dict(names):
    """Make a color dictionary to give each name a mpl color.
    """
    norm = colors.Normalize(vmin=0, vmax=1)
    m = cm.ScalarMappable(norm, cm.tab20)
    color_dict = m.to_rgba(np.linspace(0, 1, len(names)))
    color_dict = dict(zip(names, color_dict))
    return color_dict
```

[5]:
```
# Load the data
summary_ds, meta = XRSerializer.load_derived(DB_ROOT, db=DBID, key=cc.
 ↪MEAN_SCORE)
```

[6]:
```
method_to_rgba = build_color_dict(summary_ds.coords[METHOD].values.tolist())
```

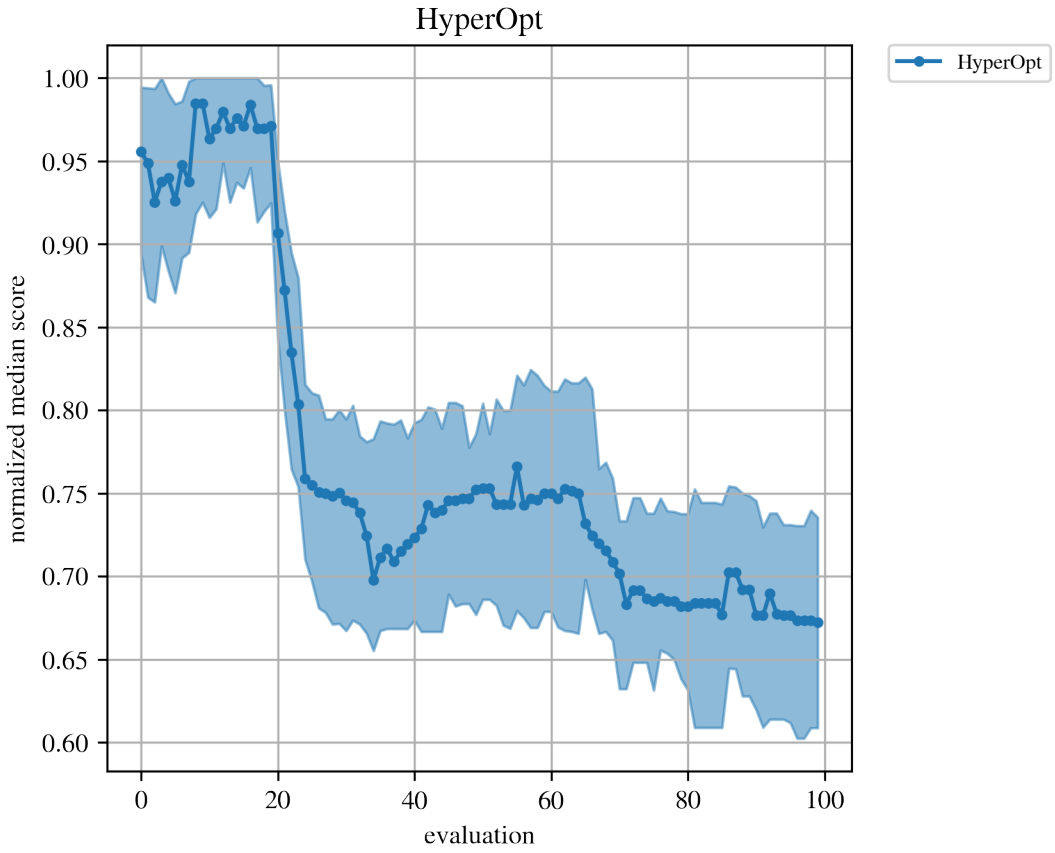
[7]:
```
# Group methods by the package behind them
method_only = lambda method_rev: method_rev.split(ARG_DELIM, 1)[0]
groups = preimage_func(method_only, summary_ds.coords[METHOD].values)
```

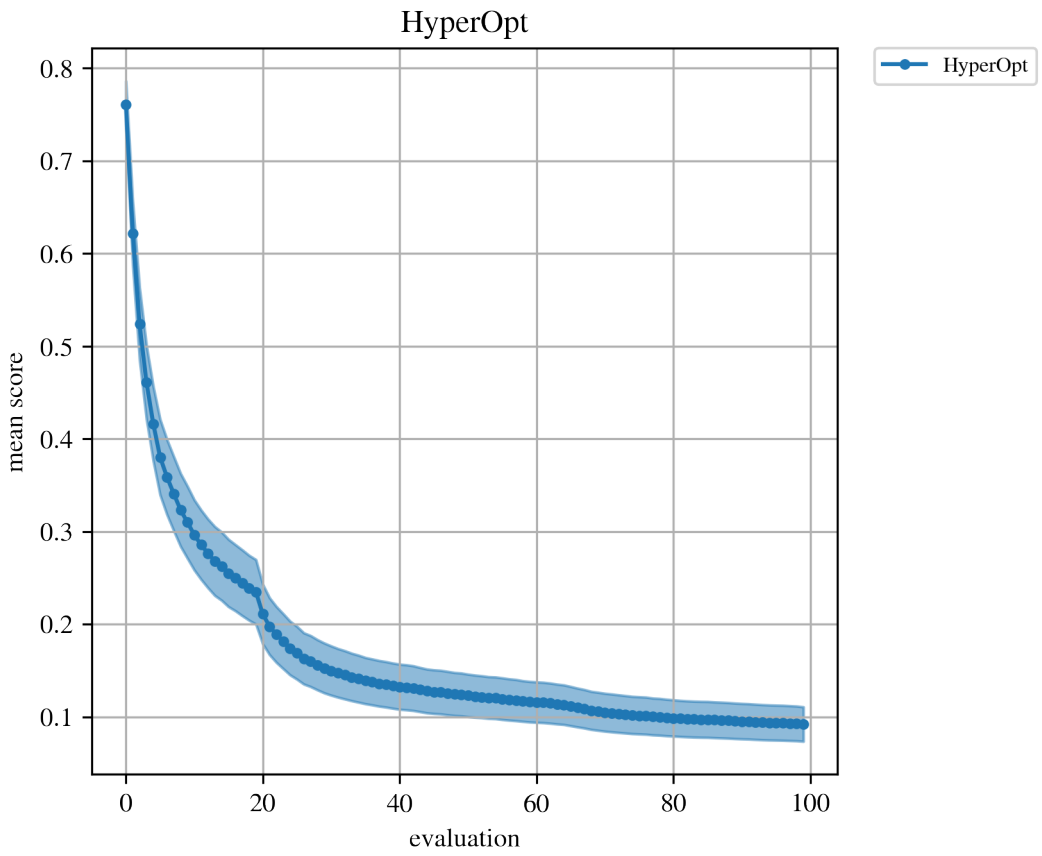
[8]:
```
# Make a plot for each package
for method_name in groups:
    plt.figure(figsize=(5, 5), dpi=300)
    for method_ver_name in groups[method_name]:
        curr_ds = summary_ds.sel({METHOD: method_ver_name})
        curr_ds.coords[ITER].values

        plt.fill_between(
            curr_ds.coords[ITER].values,
            curr_ds[cc.LB_MED].values,
            curr_ds[cc.UB_MED].values,
            color=method_to_rgba[method_ver_name],
            alpha=0.5,
        )
        plt.plot(
            curr_ds.coords[ITER].values,
            curr_ds[cc.PERF_MED].values,
            color=method_to_rgba[method_ver_name],
            label=method_name,
            marker=".",
        )
    plt.xlabel("evaluation", fontsize=10)
    plt.ylabel("normalized median score", fontsize=10)
    plt.title(method_name)
    plt.legend(fontsize=8, bbox_to_anchor=(1.05, 1), loc="upper left",␣
↪borderaxespad=0.0)
    plt.grid()

    plt.figure(figsize=(5, 5), dpi=300)
    for method_ver_name in groups[method_name]:
        curr_ds = summary_ds.sel({METHOD: method_ver_name})
        curr_ds.coords[ITER].values

        plt.fill_between(
            curr_ds.coords[ITER].values,
            curr_ds[cc.LB_MEAN].values,
            curr_ds[cc.UB_MEAN].values,
            color=method_to_rgba[method_ver_name],
            alpha=0.5,
        )
        plt.plot(
            curr_ds.coords[ITER].values,
            curr_ds[cc.PERF_MEAN].values,
```

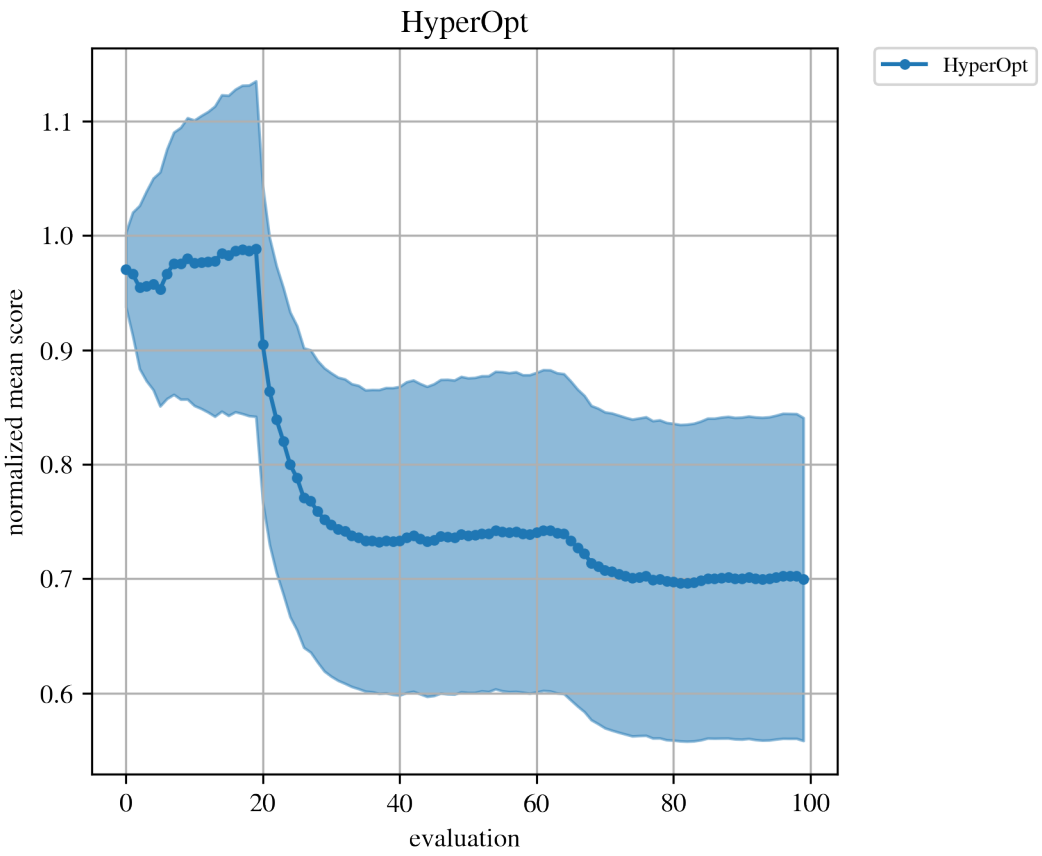
```
            color=method_to_rgba[method_ver_name],
            label=method_name,
            marker=".",
        )
    plt.xlabel("evaluation", fontsize=10)
    plt.ylabel("mean score", fontsize=10)
    plt.title(method_name)
    plt.legend(fontsize=8, bbox_to_anchor=(1.05, 1), loc="upper left",␣
↪borderaxespad=0.0)
    plt.grid()

    plt.figure(figsize=(5, 5), dpi=300)
    for method_ver_name in groups[method_name]:
        curr_ds = summary_ds.sel({METHOD: method_ver_name})
        curr_ds.coords[ITER].values

        plt.fill_between(
            curr_ds.coords[ITER].values,
            curr_ds[cc.LB_NORMED_MEAN].values,
            curr_ds[cc.UB_NORMED_MEAN].values,
            color=method_to_rgba[method_ver_name],
            alpha=0.5,
        )
        plt.plot(
            curr_ds.coords[ITER].values,
            curr_ds[cc.NORMED_MEAN].values,
            color=method_to_rgba[method_ver_name],
            label=method_name,
            marker=".",
        )
    plt.xlabel("evaluation", fontsize=10)
    plt.ylabel("normalized mean score", fontsize=10)
    plt.title(method_name)
    plt.legend(fontsize=8, bbox_to_anchor=(1.05, 1), loc="upper left",␣
↪borderaxespad=0.0)
    plt.grid()
```

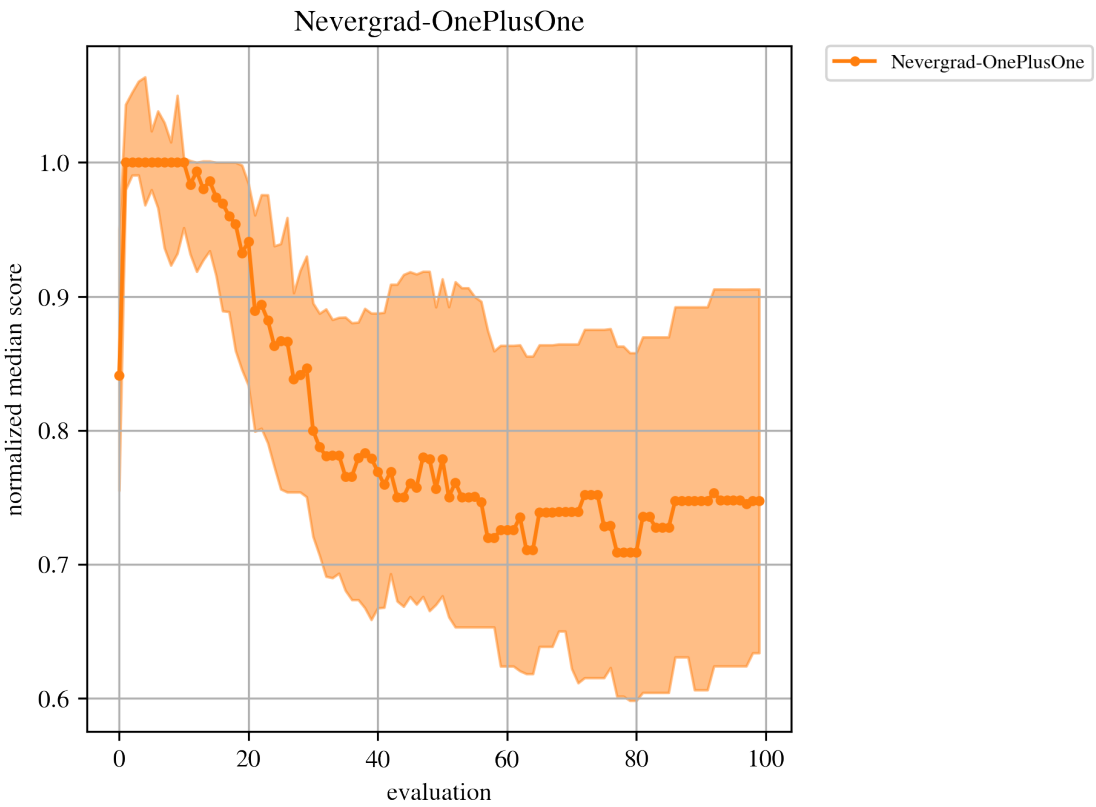
```
/Users/ryan.turner/envs/bobm_ipynb/lib/python3.6/site-
packages/ipykernel_launcher.py:53: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface
(`matplotlib.pyplot.figure`) are retained until explicitly closed and may
consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
/Users/ryan.turner/envs/bobm_ipynb/lib/python3.6/site-
packages/ipykernel_launcher.py:3: RuntimeWarning: More than 20 figures have been
opened. Figures created through the pyplot interface
(`matplotlib.pyplot.figure`) are retained until explicitly closed and may
```

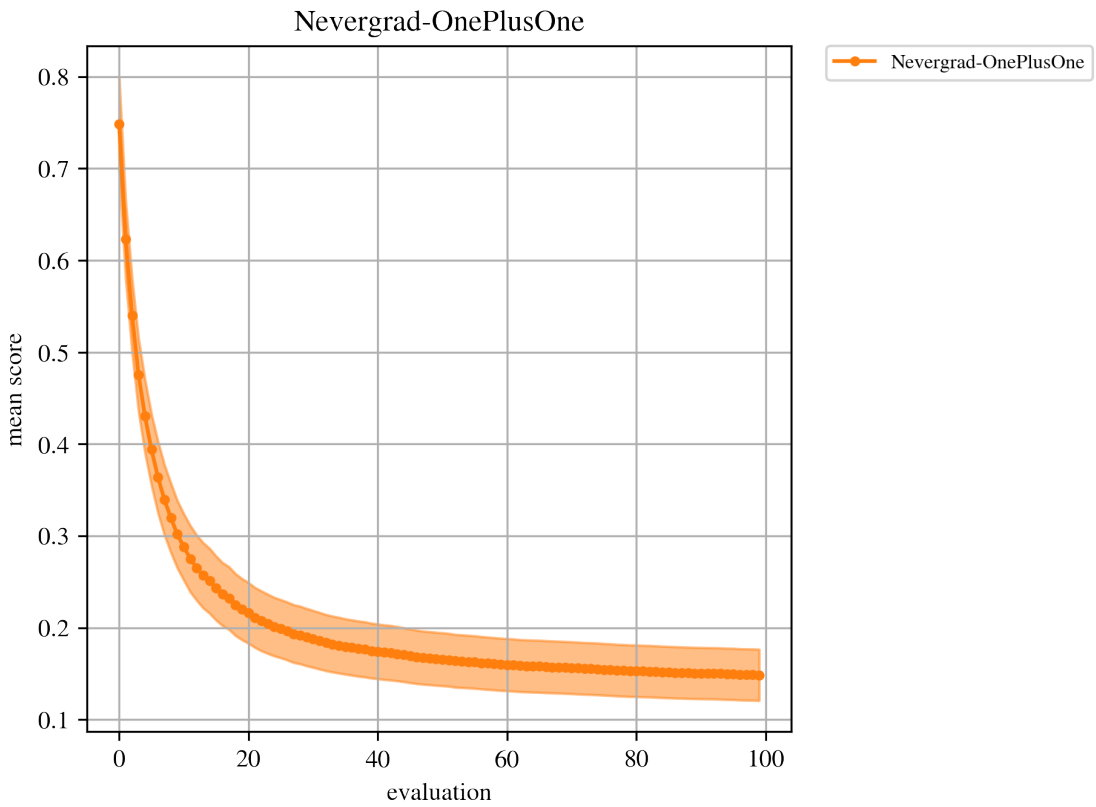
```
consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  This is separate from the ipykernel package so we can avoid doing imports
until
/Users/ryan.turner/envs/bobm_ipynb/lib/python3.6/site-
packages/ipykernel_launcher.py:28: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface
(`matplotlib.pyplot.figure`) are retained until explicitly closed and may
consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
```

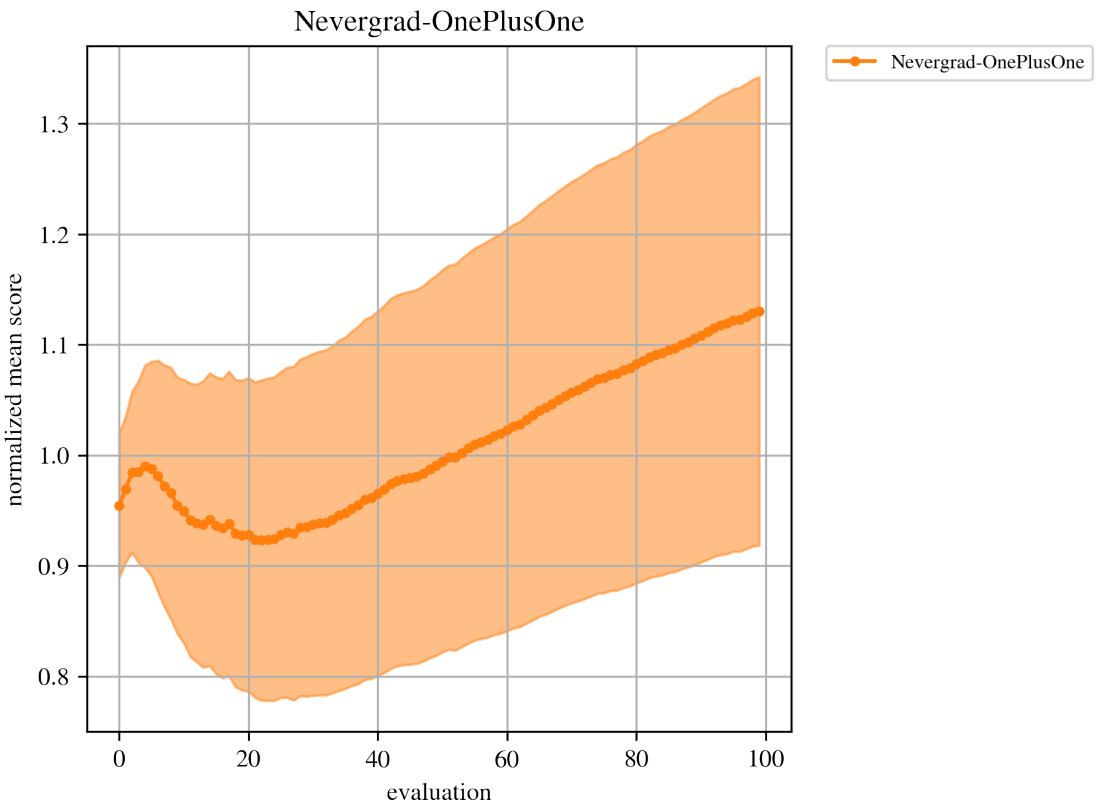


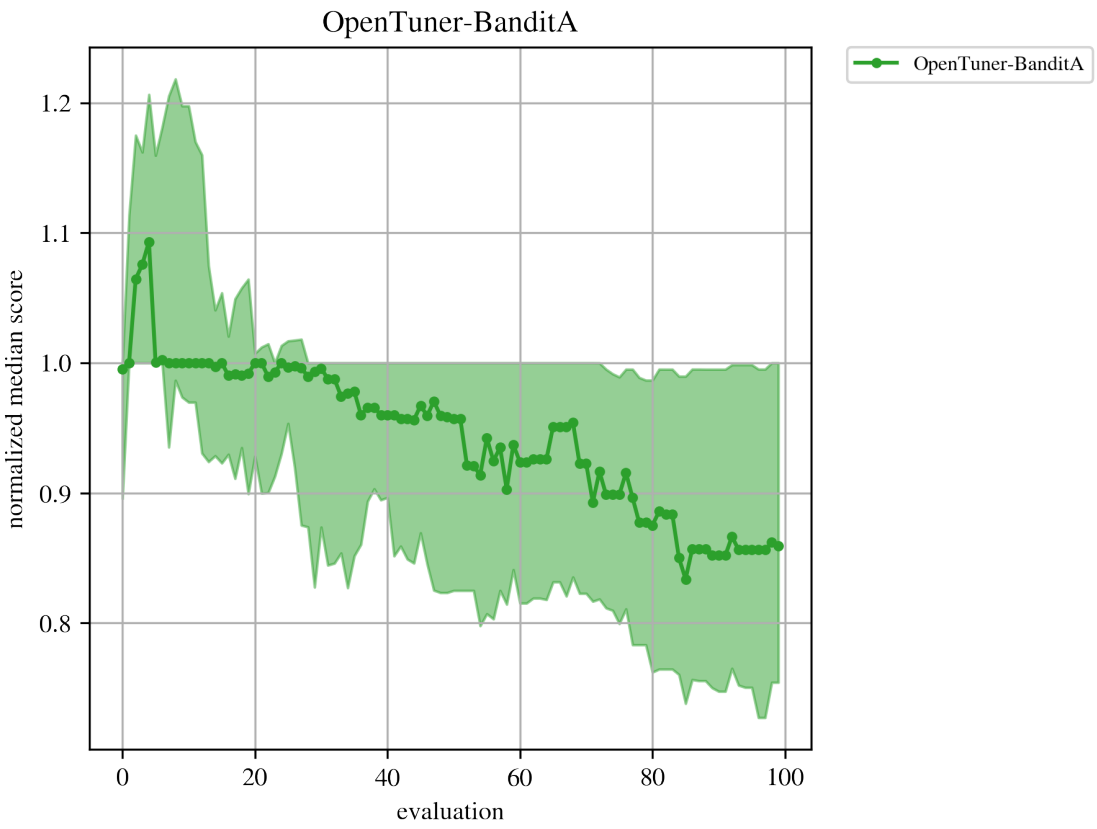

HyperOpt
HyperOpt
mean score
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100
evaluation

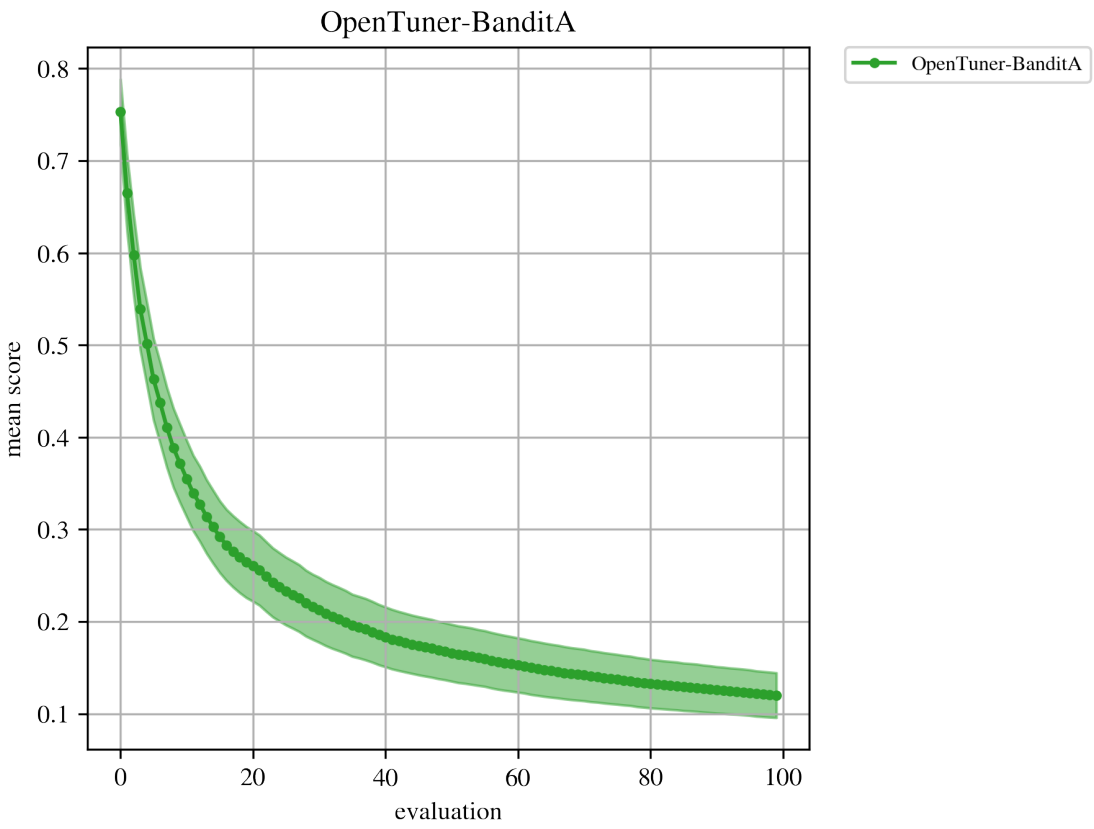

HyperOpt
HyperOpt
normalized mean score
1.1
1.0
0.9
0.8
0.7
0.6
0
20
40
60
80
100
evaluation

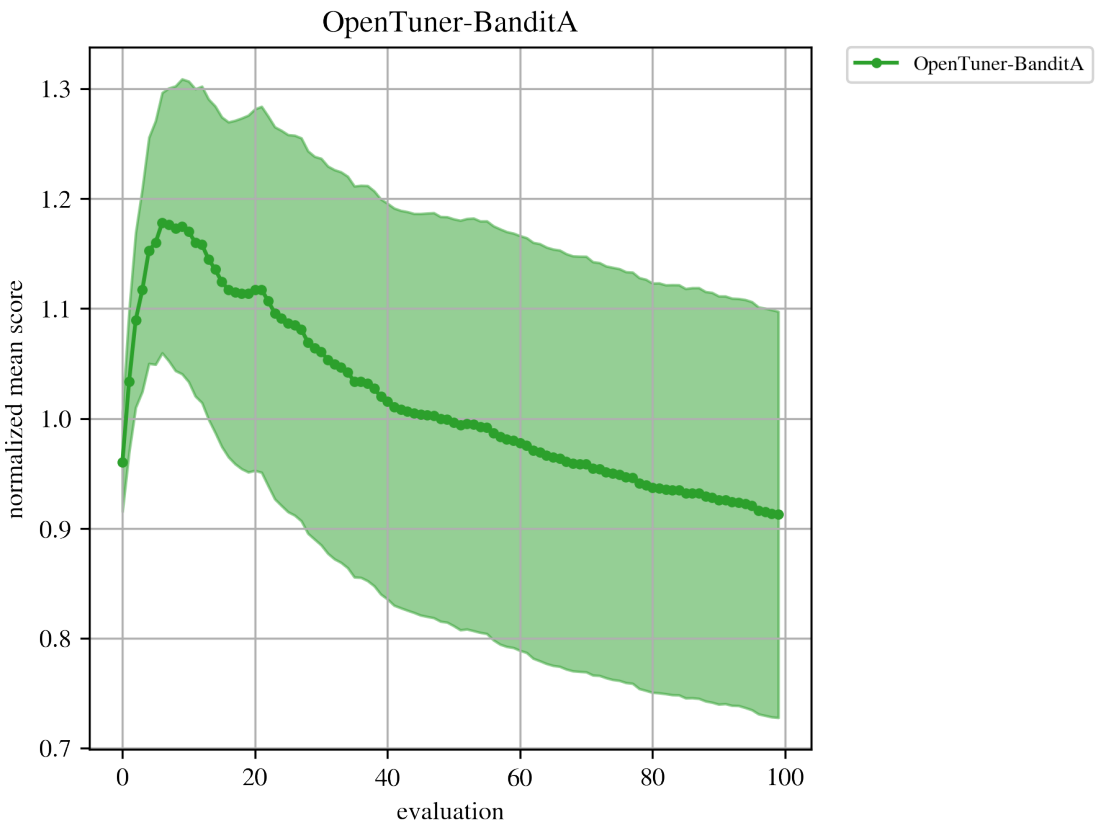

Nevergrad-OnePlusOne
Nevergrad-OnePlusOne
normalized median score
1.0
0.9
0.8
0.7
0.6
0
20
40
60
80
100
evaluation

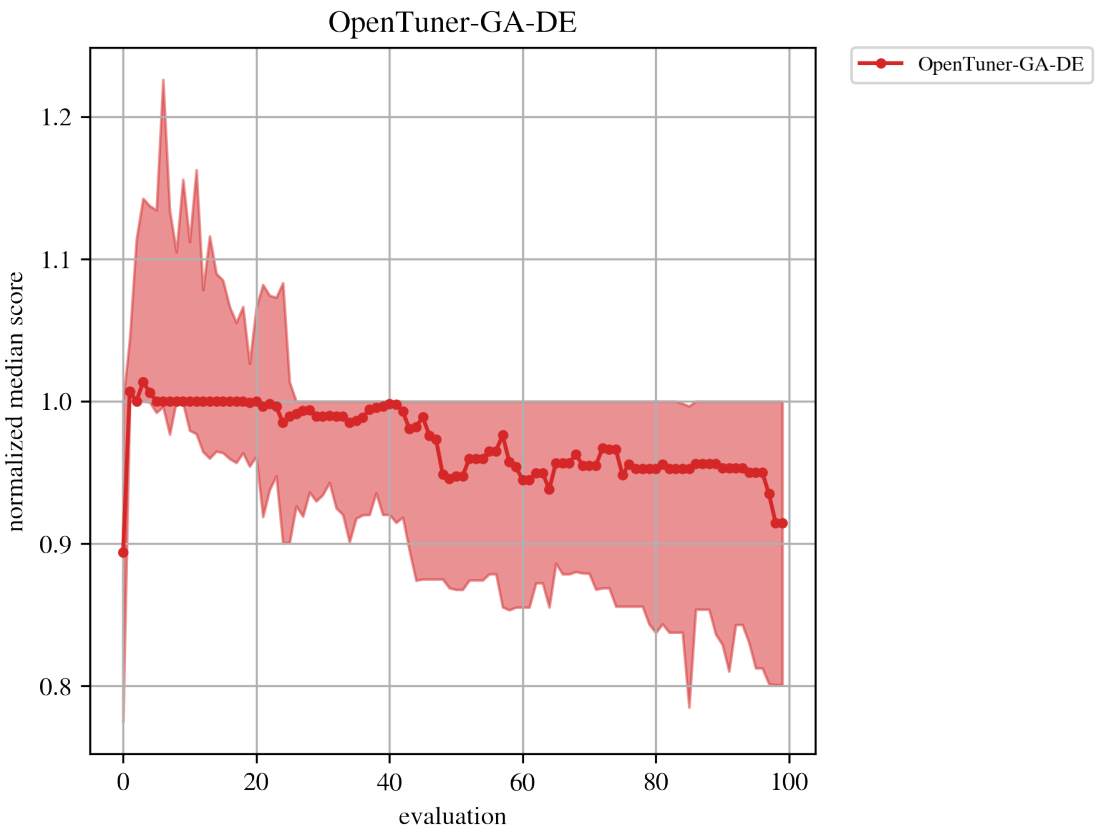

Nevergrad-OnePlusOne
Nevergrad-OnePlusOne
mean score
evaluation
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100

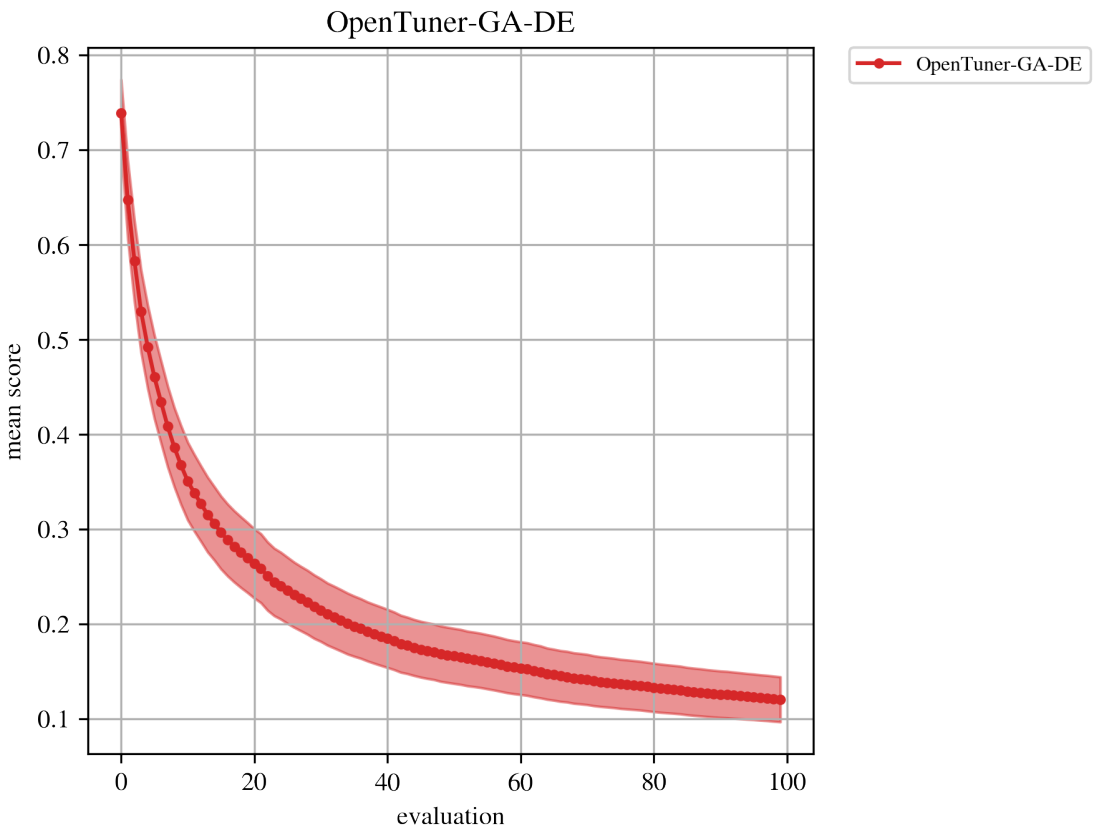

Nevergrad-OnePlusOne
Nevergrad-OnePlusOne
normalized mean score
1.3
1.2
1.1
1.0
0.9
0.8
0
20
40
60
80
100
evaluation

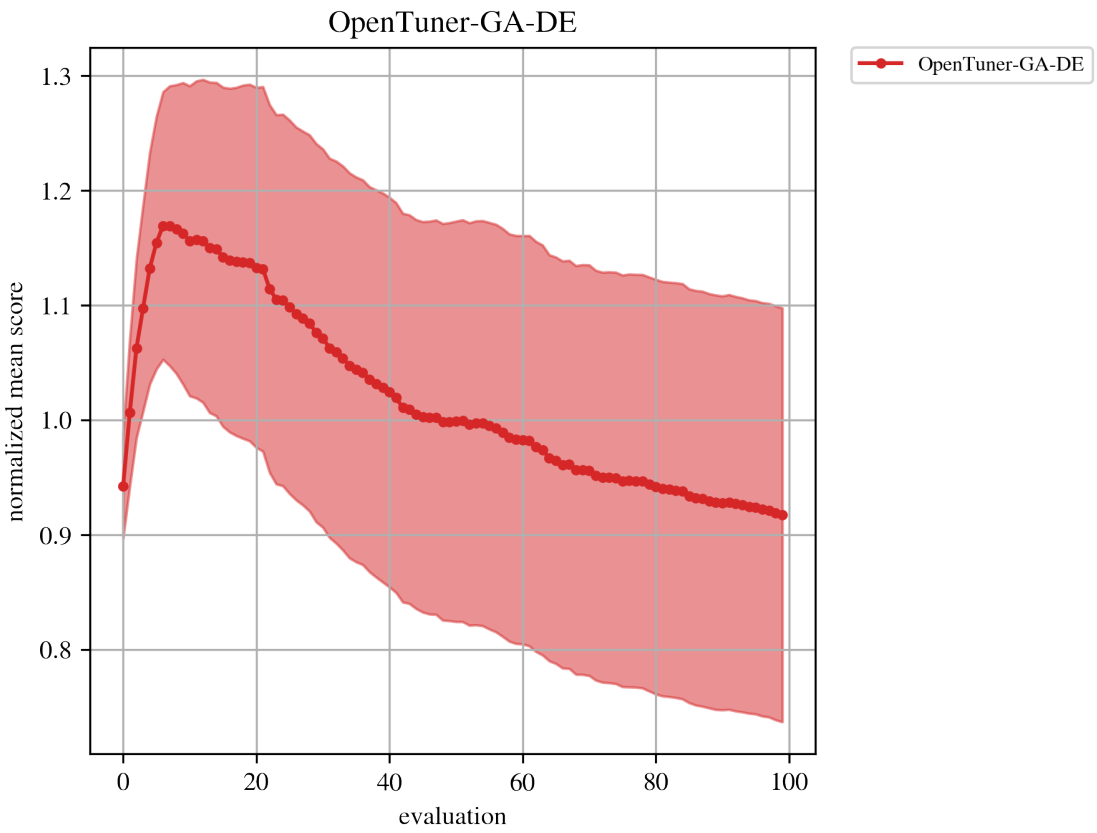

OpenTuner-BanditA
OpenTuner-BanditA
normalized median score
1.2
1.1
1.0
0.9
0.8
0
20
40
60
80
100
evaluation

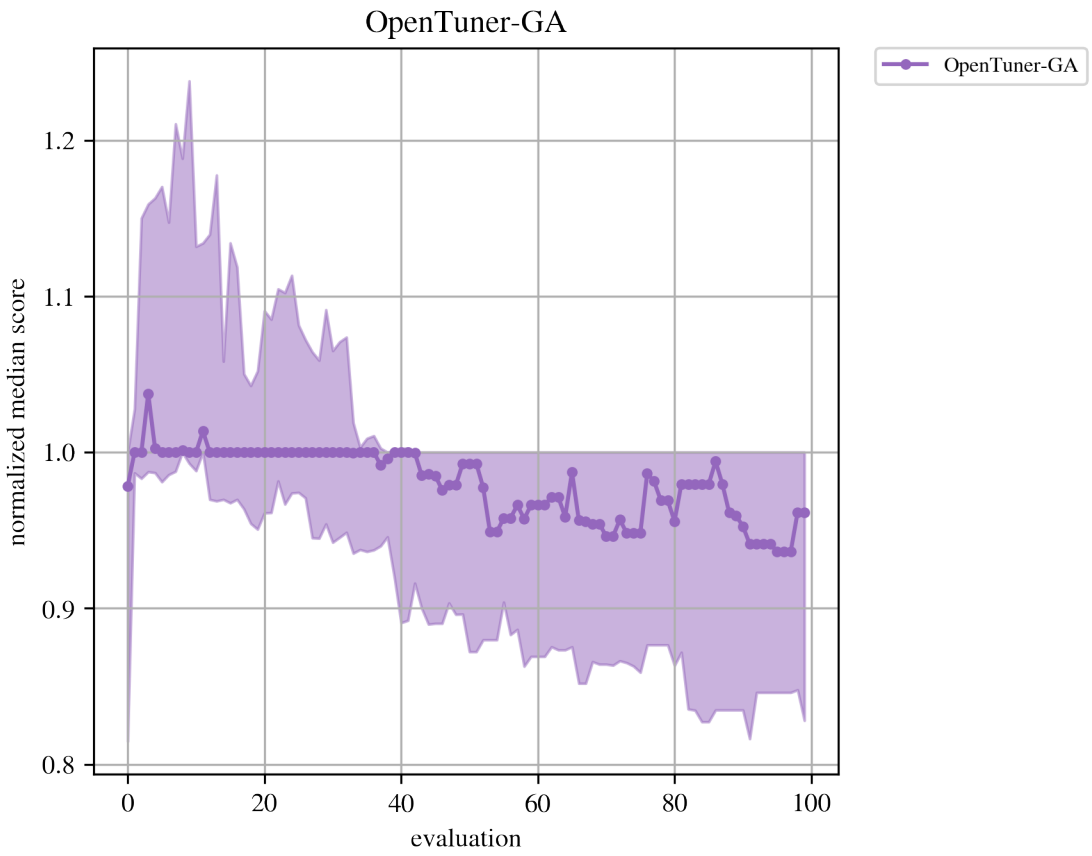

OpenTuner-BanditA
mean score
evaluation
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100
OpenTuner-BanditA

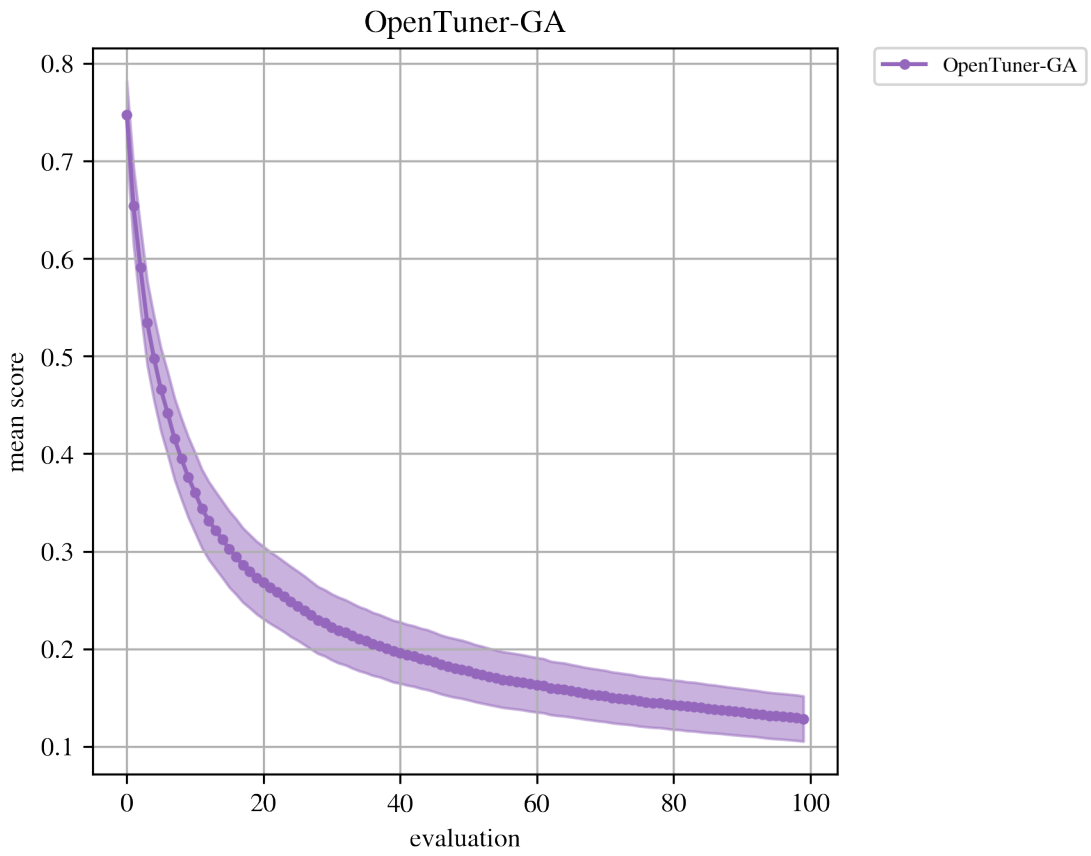

OpenTuner-BanditA
OpenTuner-BanditA
normalized mean score
1.3
1.2
1.1
1.0
0.9
0.8
0.7
0
20
40
60
80
100
evaluation


OpenTuner-GA-DE
OpenTuner-GA-DE
normalized median score
1.2
1.1
1.0
0.9
0.8
0
20
40
60
80
100
evaluation

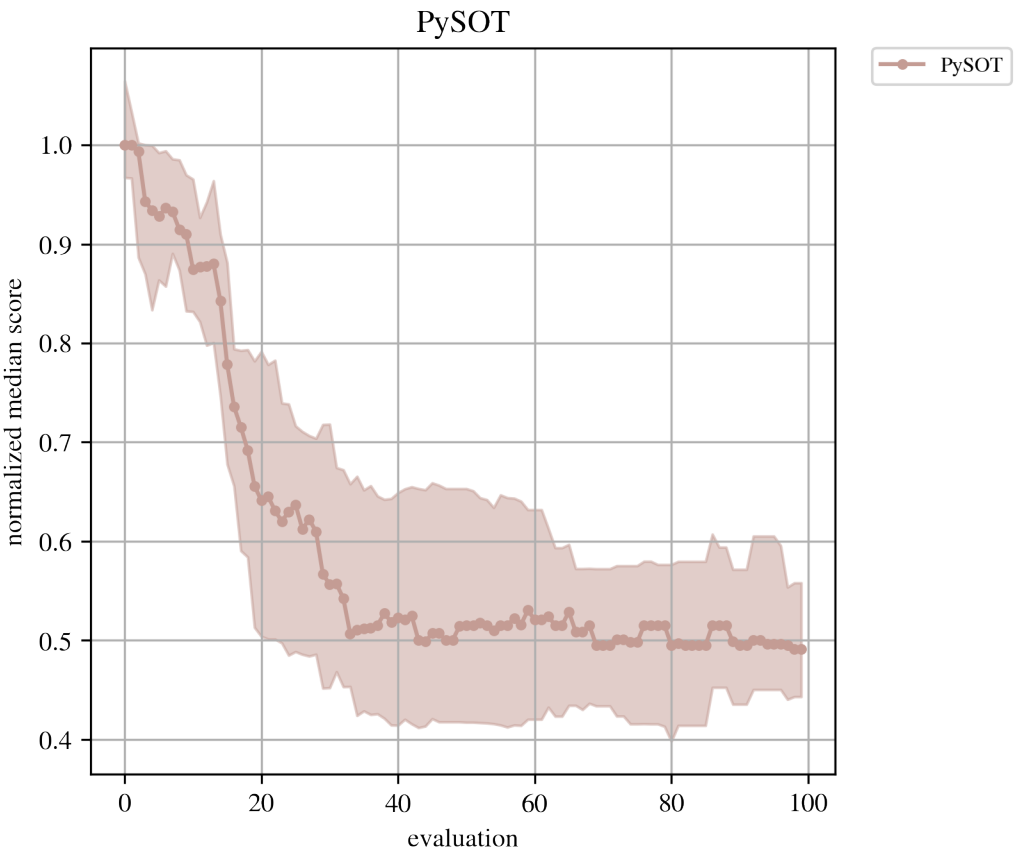

OpenTuner-GA-DE
mean score
evaluation
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100
OpenTuner-GA-DE

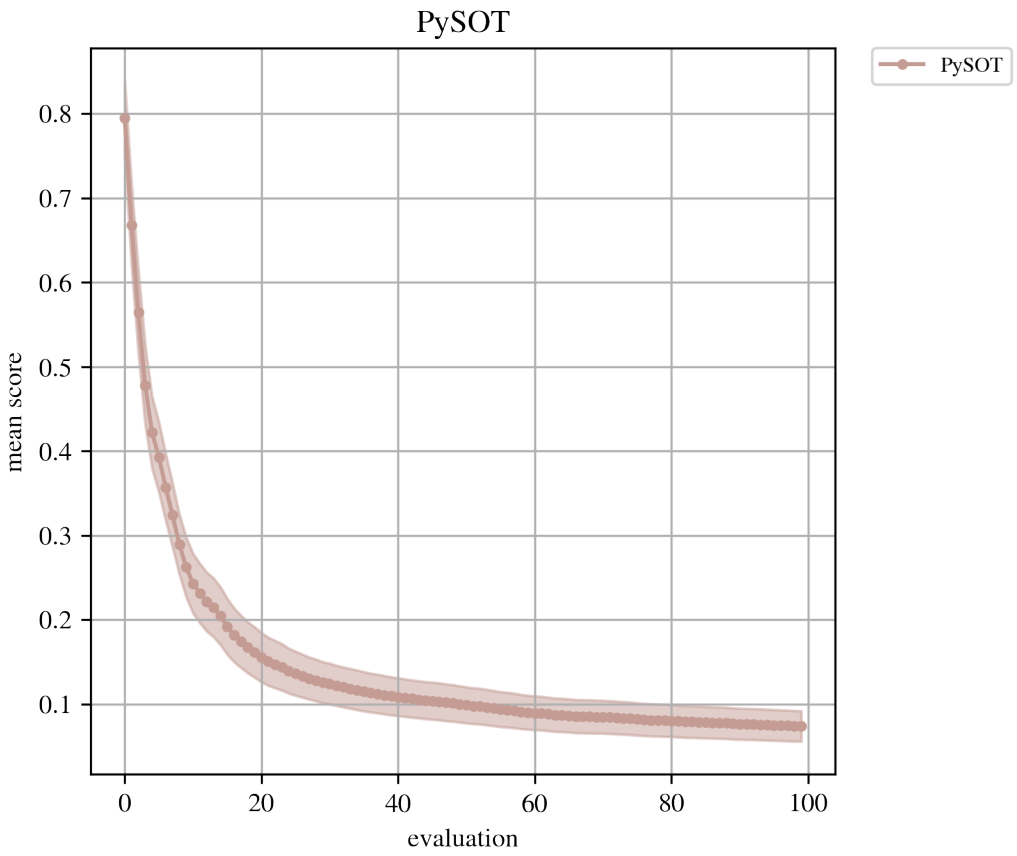

OpenTuner-GA-DE
OpenTuner-GA-DE
normalized mean score
1.3
1.2
1.1
1.0
0.9
0.8
0
20
40
60
80
100
evaluation

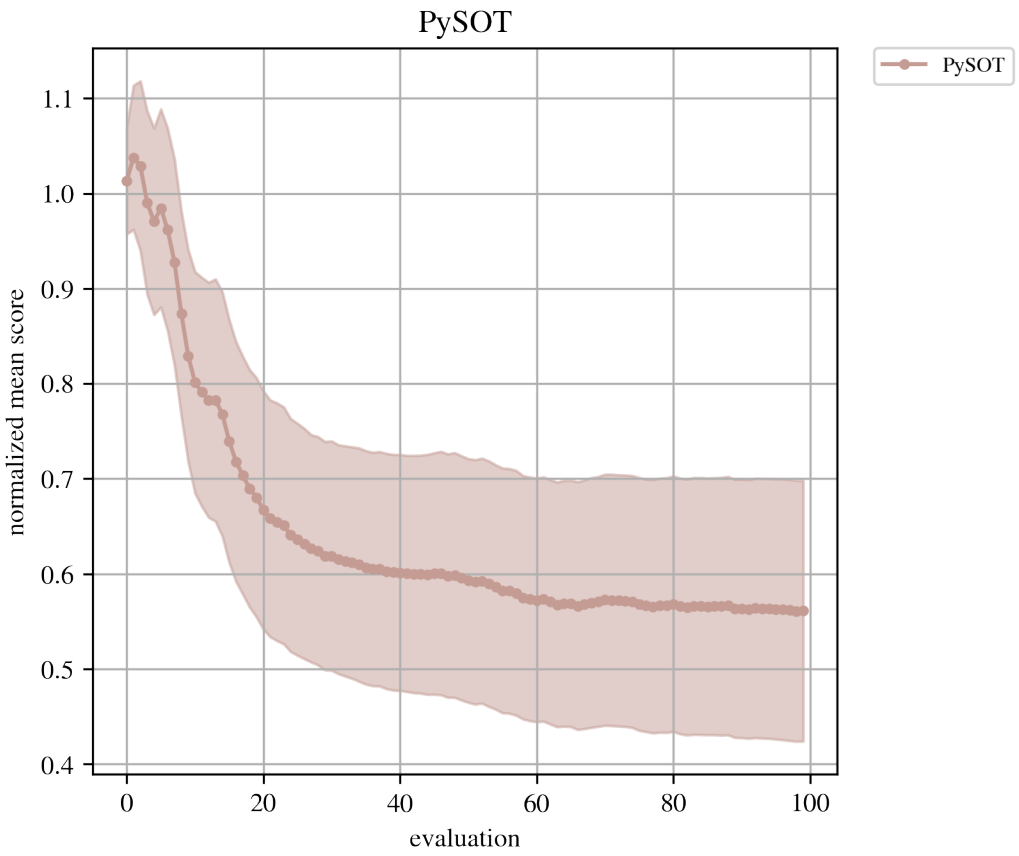

OpenTuner-GA
OpenTuner-GA
normalized median score
1.2
1.1
1.0
0.9
0.8
0
20
40
60
80
100
evaluation

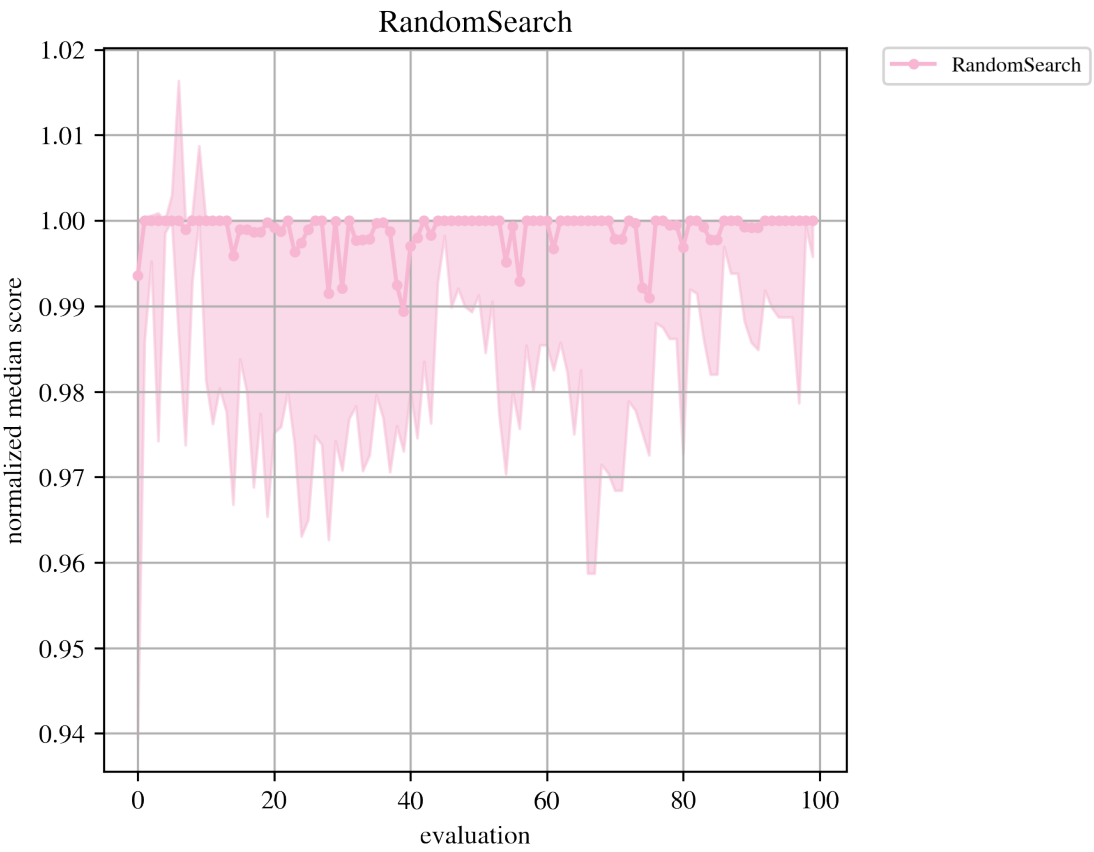

OpenTuner-GA
mean score
evaluation
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100
OpenTuner-GA

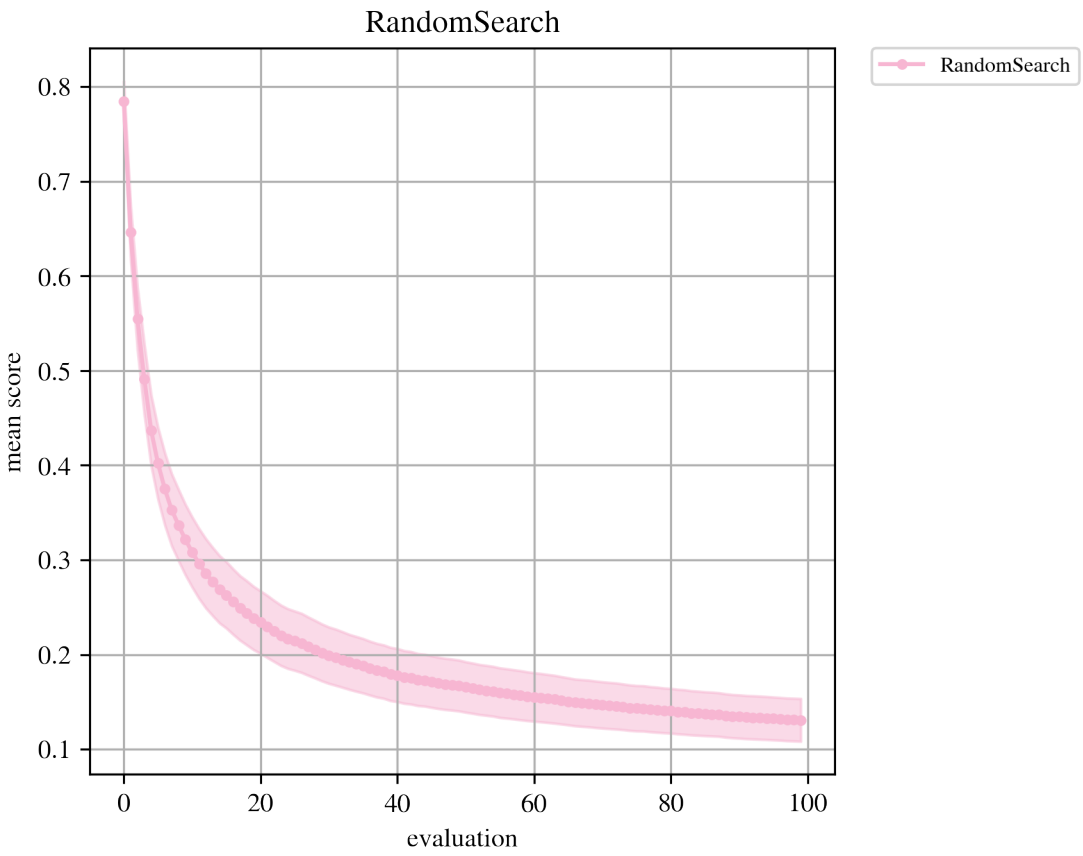

OpenTuner-GA
OpenTuner-GA
normalized mean score
1.3
1.2
1.1
1.0
0.9
0.8
0
20
40
60
80
100
evaluation

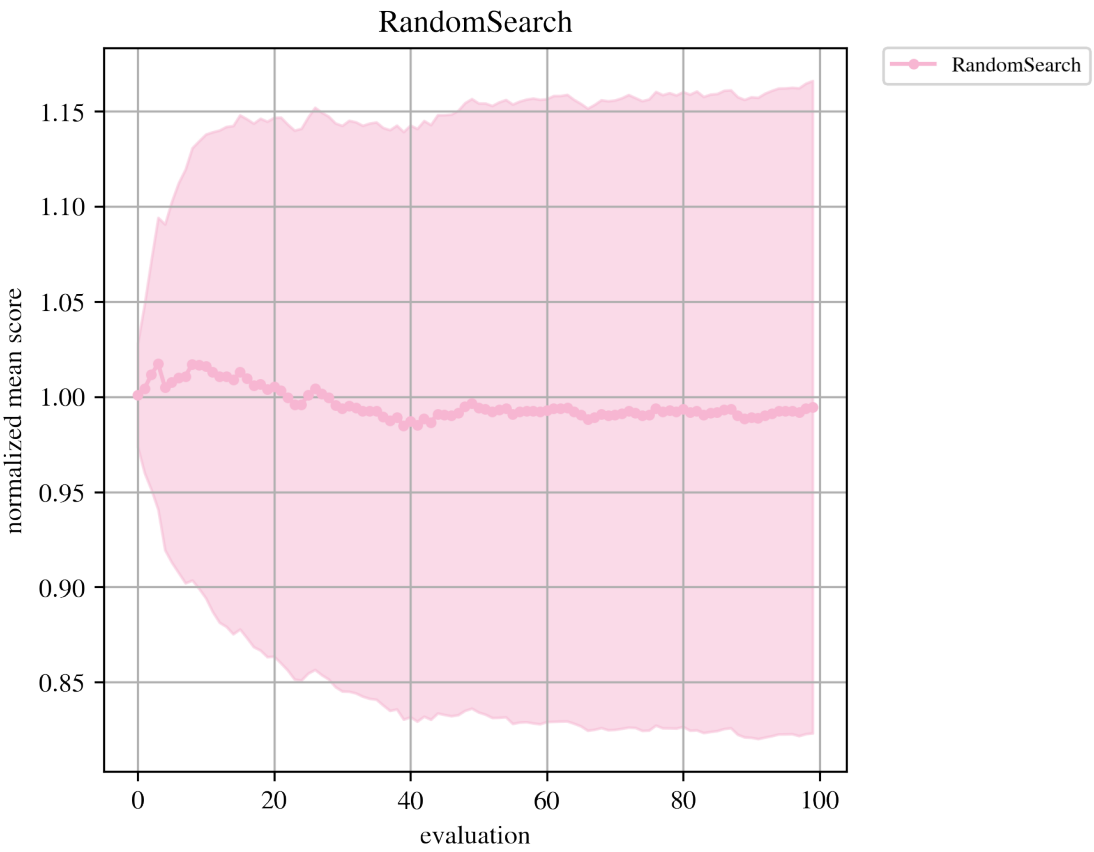

PySOT
PySOT
normalized median score
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
20
40
60
80
100
evaluation


PySOT
PySOT
mean score
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100
evaluation

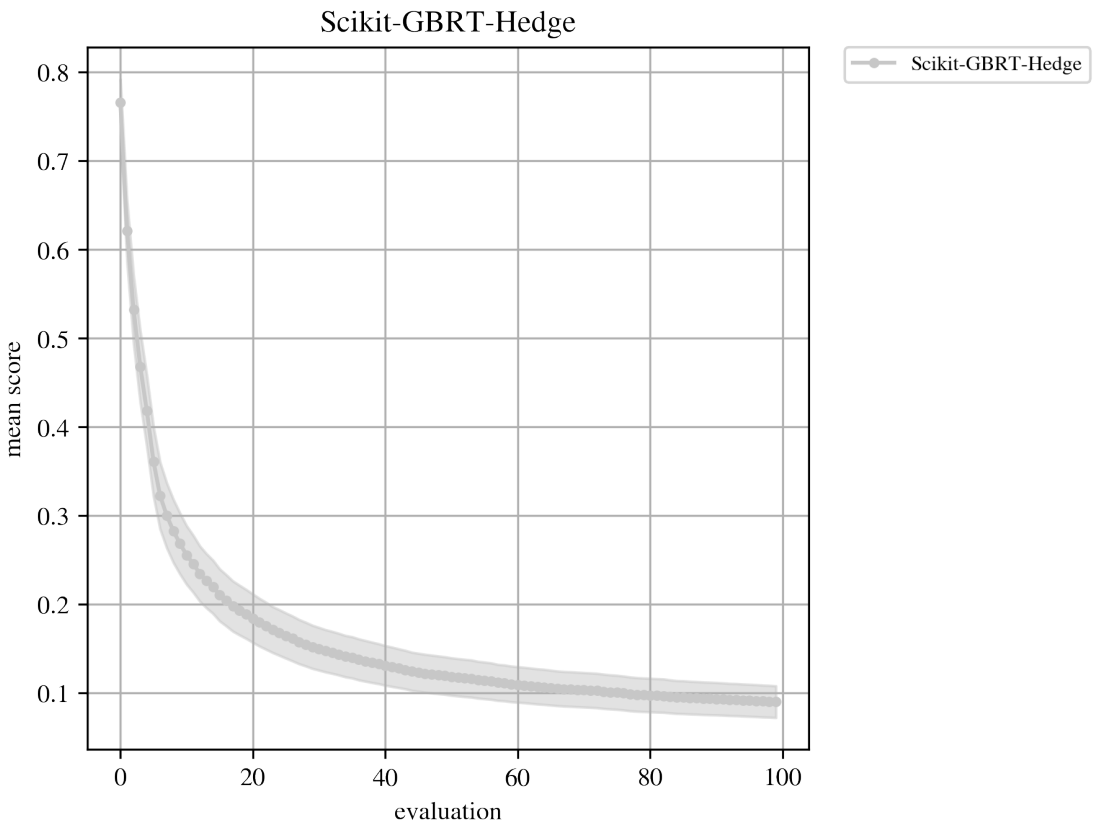

PySOT
PySOT
normalized mean score
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
20
40
60
80
100
evaluation

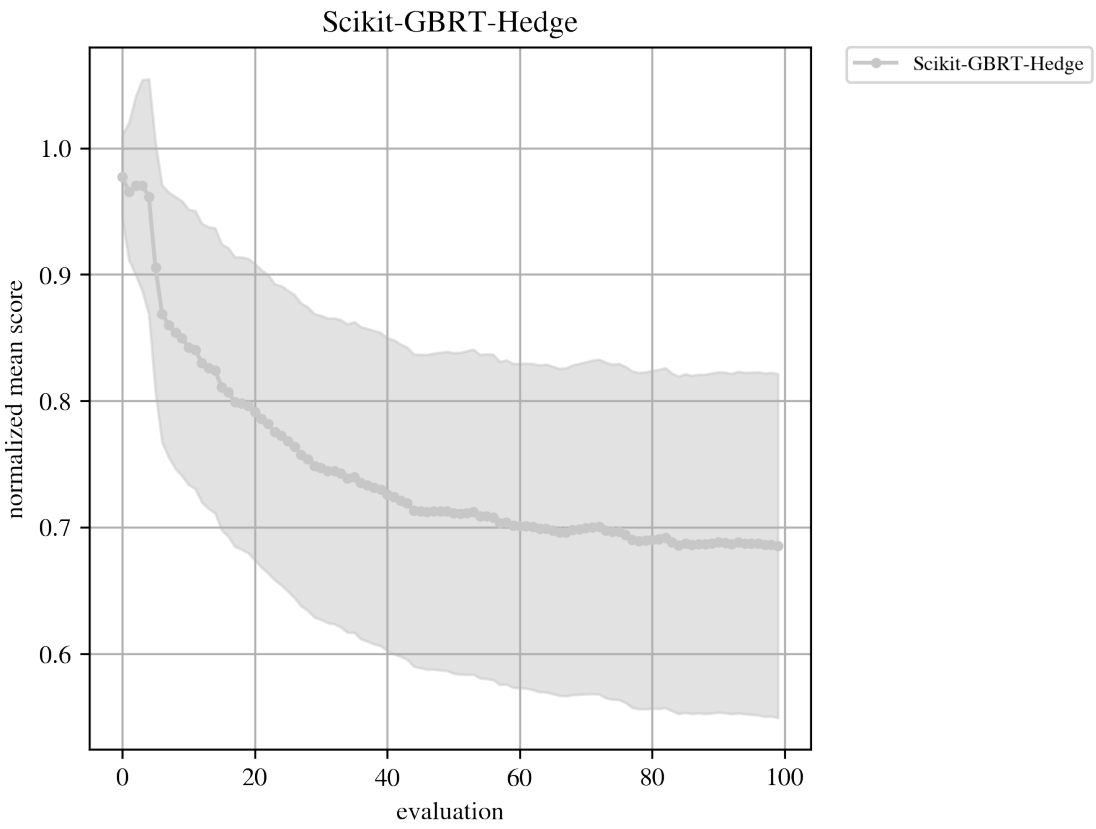

RandomSearch
RandomSearch
normalized median score
1.02
1.01
1.00
0.99
0.98
0.97
0.96
0.95
0.94
0
20
40
60
80
100
evaluation

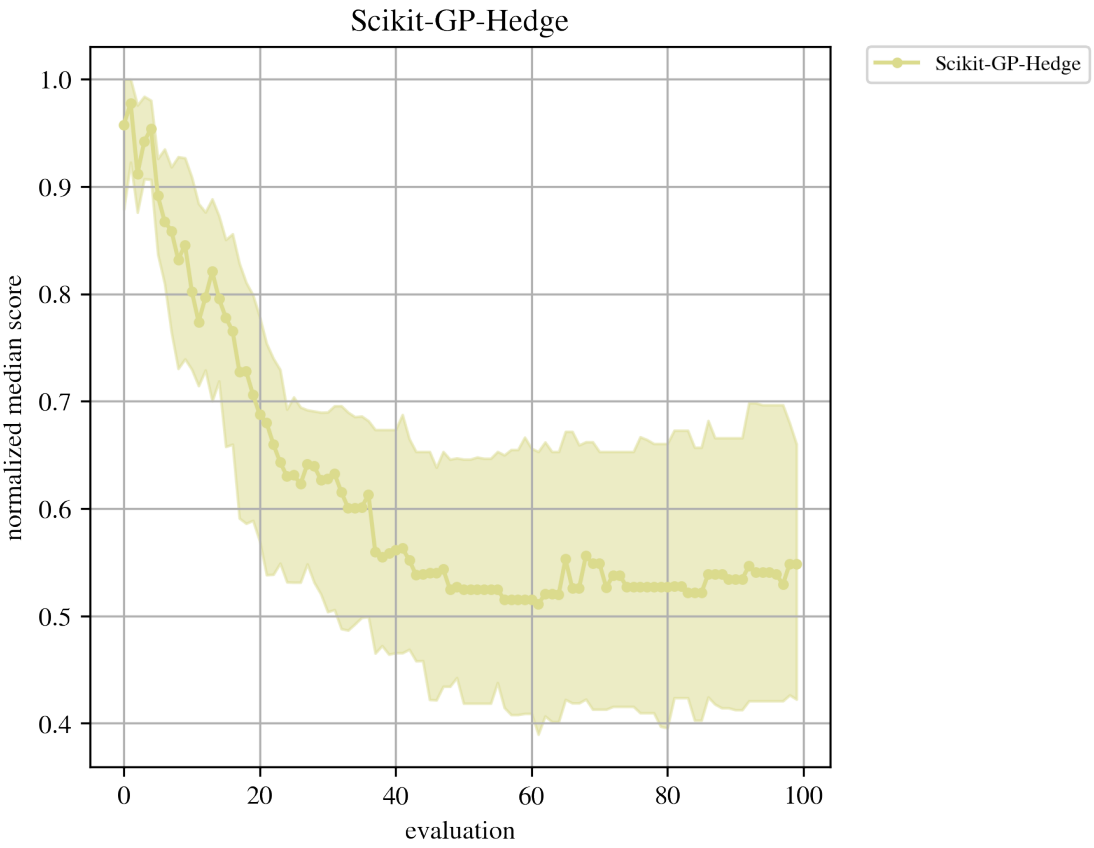

RandomSearch
RandomSearch
mean score
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100
evaluation

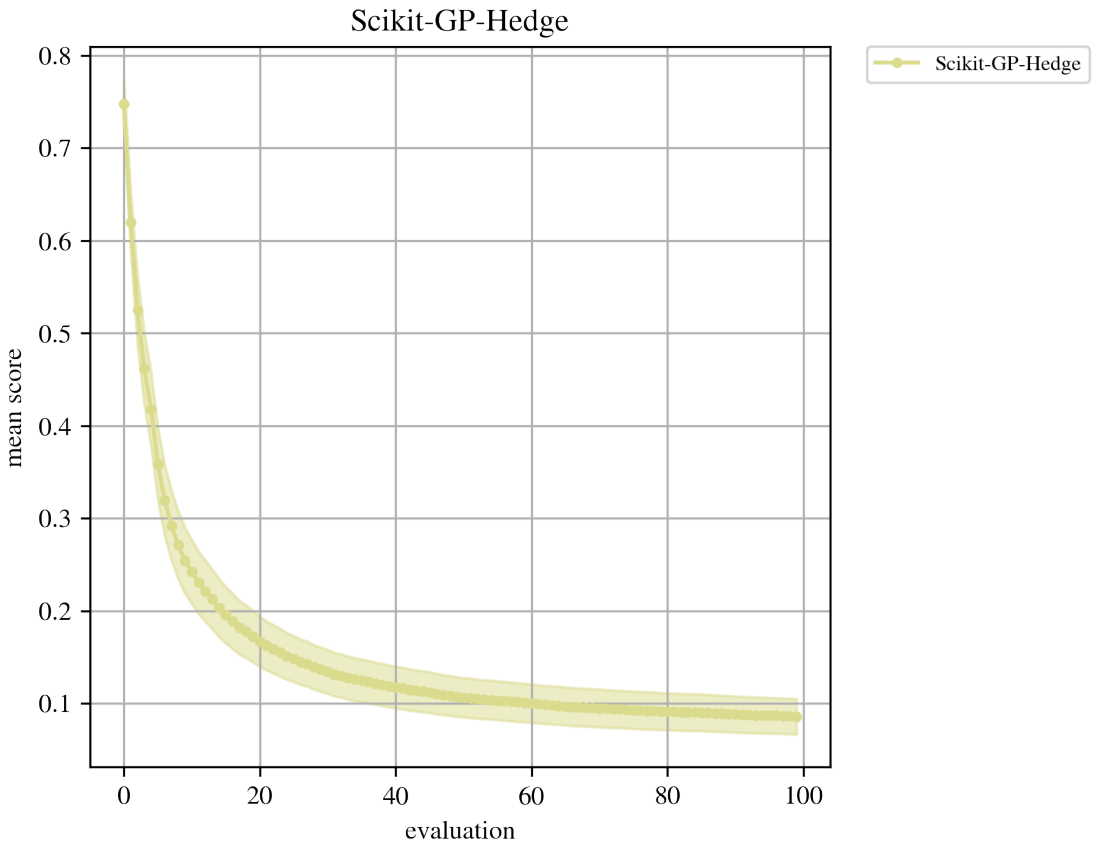

RandomSearch
RandomSearch
normalized mean score
1.15
1.10
1.05
1.00
0.95
0.90
0.85
0
20
40
60
80
100
evaluation

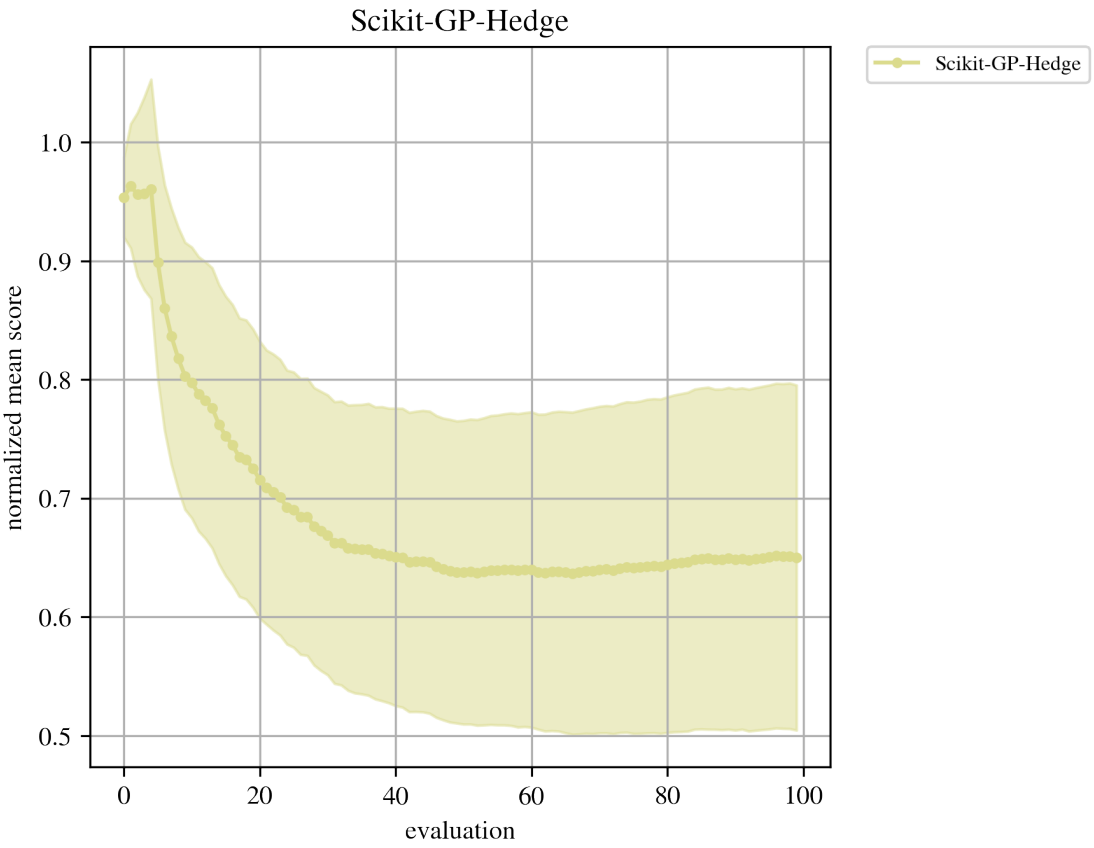

Scikit-GBRT-Hedge
Scikit-GBRT-Hedge
normalized median score
1.0
0.9
0.8
0.7
0.6
0.5
0
20
40
60
80
100
evaluation

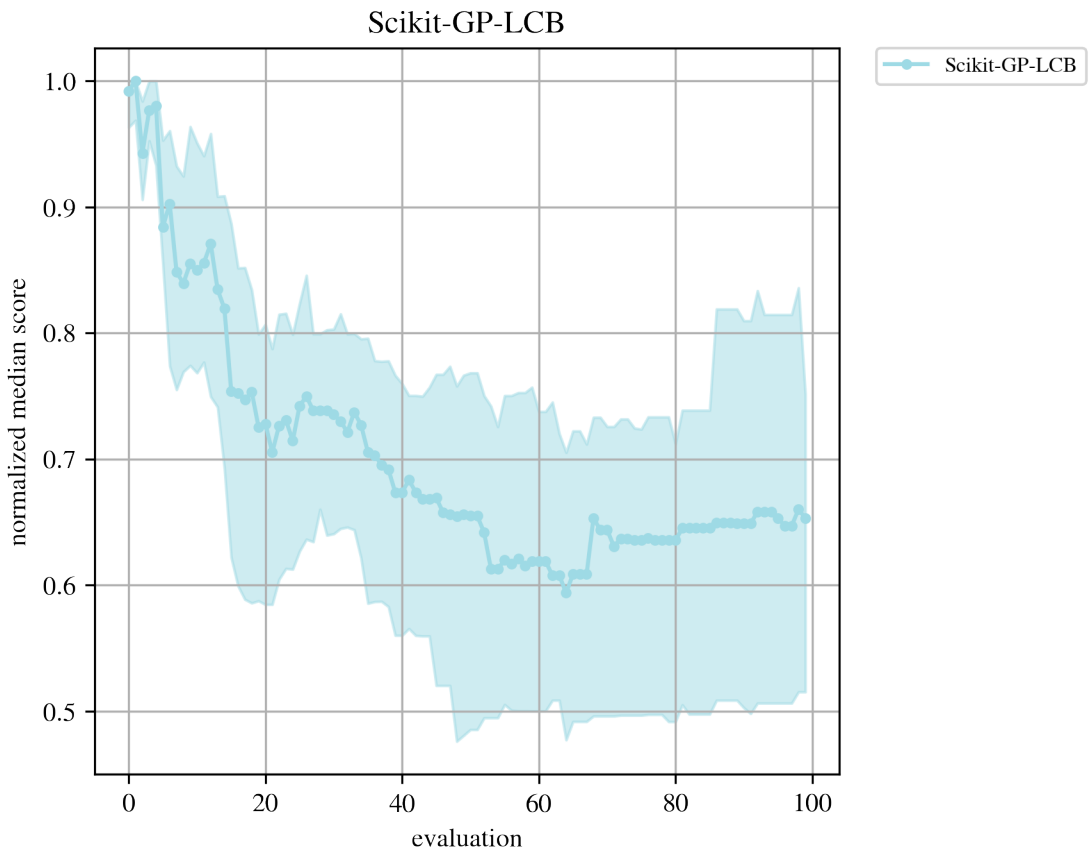

Scikit-GBRT-Hedge
Scikit-GBRT-Hedge
mean score
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100
evaluation

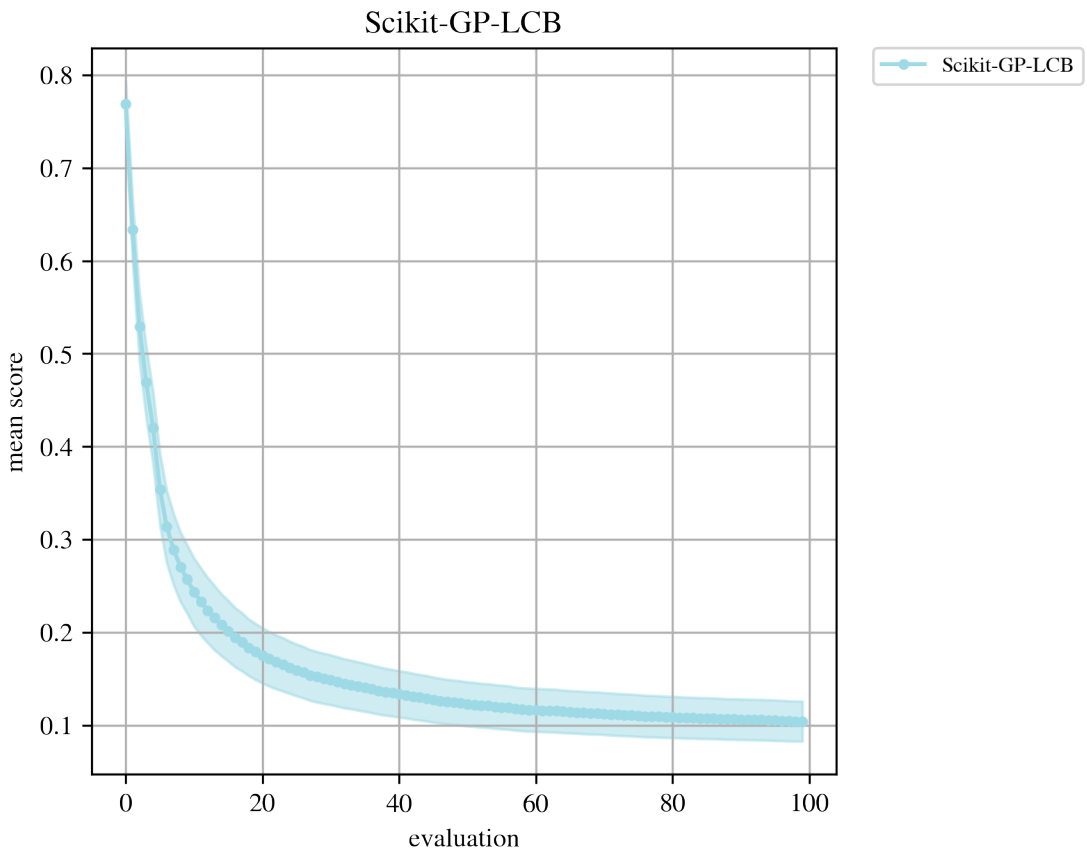

Scikit-GBRT-Hedge
Scikit-GBRT-Hedge
normalized mean score
1.0
0.9
0.8
0.7
0.6
0
20
40
60
80
100
evaluation

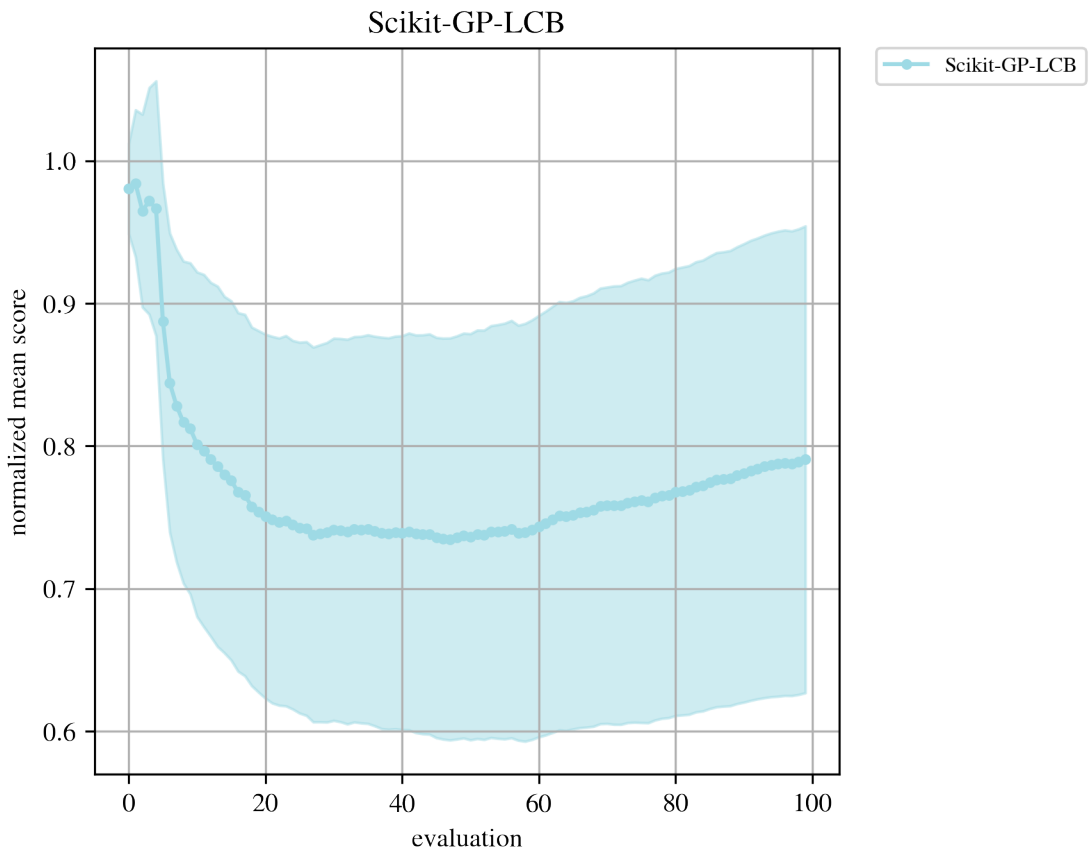

Scikit-GP-Hedge
Scikit-GP-Hedge
normalized median score
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
20
40
60
80
100
evaluation


Scikit-GP-Hedge
Scikit-GP-Hedge
mean score
evaluation
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100


Scikit-GP-Hedge
Scikit-GP-Hedge
normalized mean score
1.0
0.9
0.8
0.7
0.6
0.5
0
20
40
60
80
100
evaluation


Scikit-GP-LCB
Scikit-GP-LCB
normalized median score
1.0
0.9
0.8
0.7
0.6
0.5
0
20
40
60
80
100
evaluation

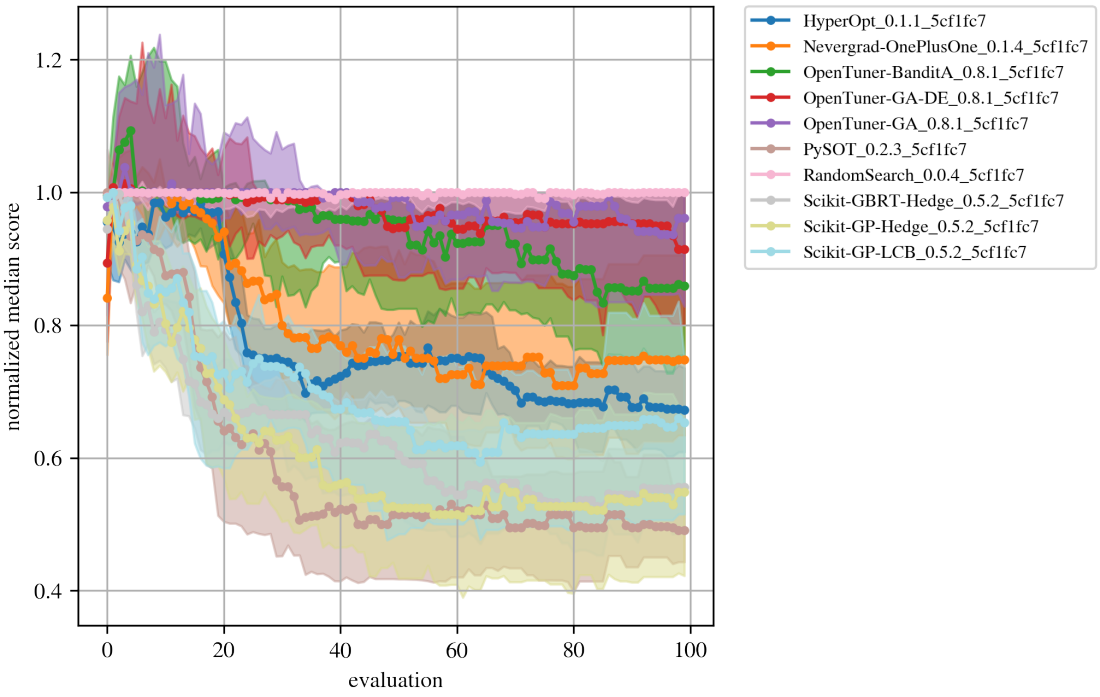

Scikit-GP-LCB
Scikit-GP-LCB
mean score
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100
evaluation

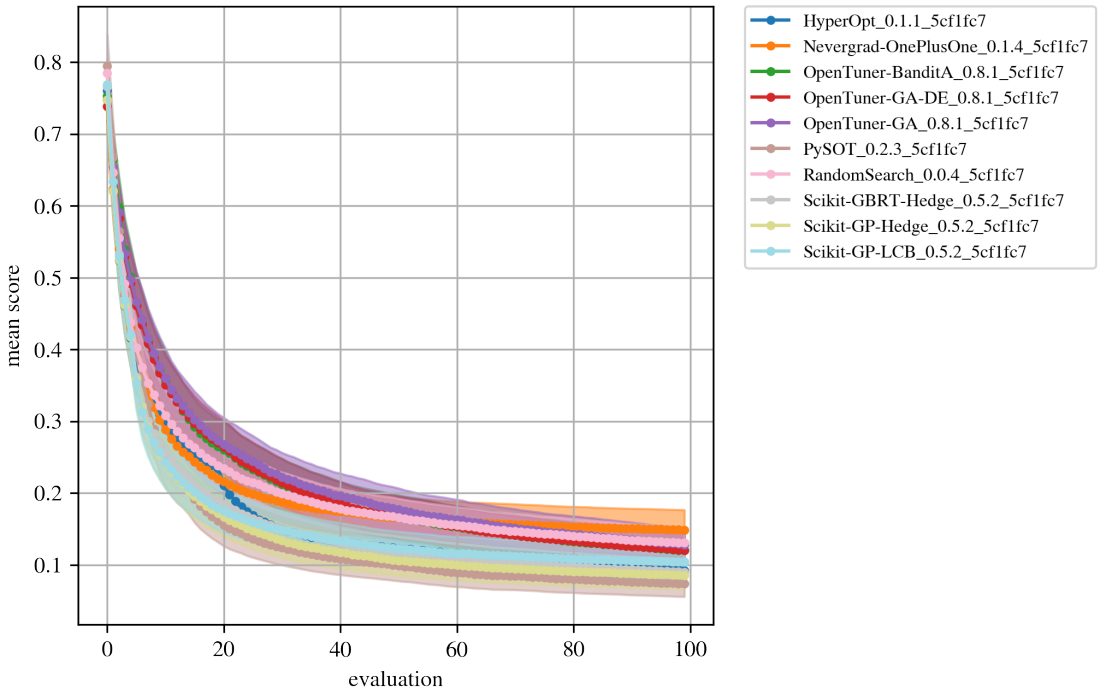


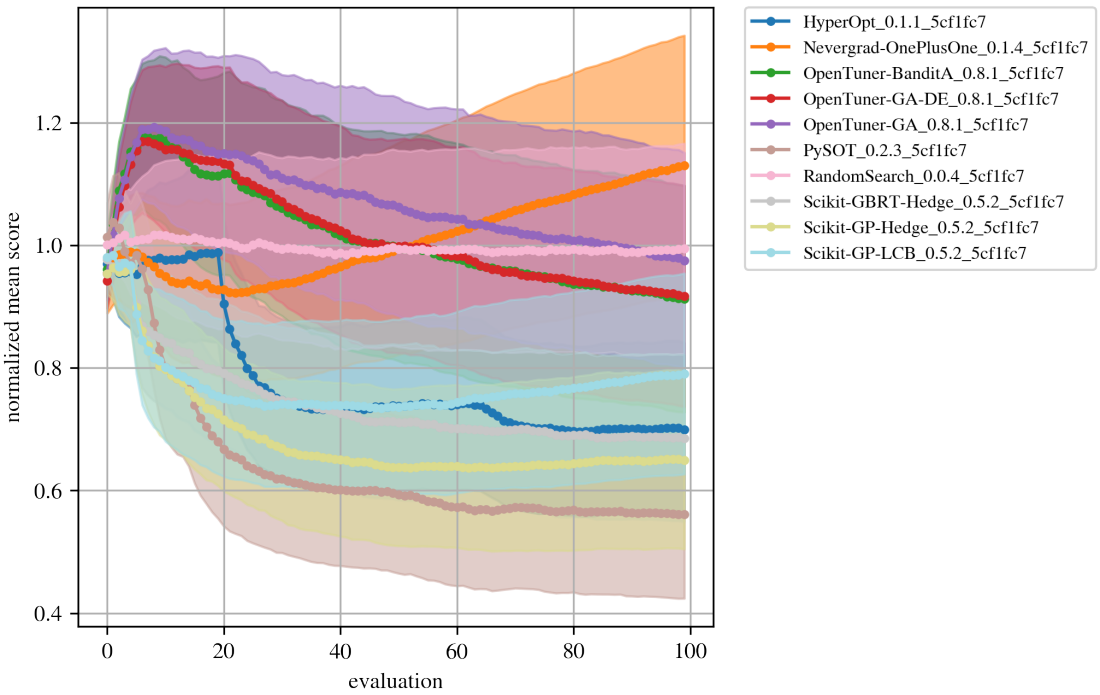
[9]:
```
# Make the summary plot
plt.figure(figsize=(5, 5), dpi=300)
for method_ver_name in summary_ds.coords[METHOD].values:
    curr_ds = summary_ds.sel({METHOD: method_ver_name})
    curr_ds.coords[ITER].values

    plt.fill_between(
        curr_ds.coords[ITER].values,
        curr_ds[cc.LB_MED].values,
        curr_ds[cc.UB_MED].values,
        color=method_to_rgba[method_ver_name],
        alpha=0.5,
    )
    plt.plot(
        curr_ds.coords[ITER].values,
        curr_ds[cc.PERF_MED].values,
        color=method_to_rgba[method_ver_name],
        label=method_ver_name,
        marker=".",
    )
```

```
plt.xlabel("evaluation", fontsize=10)
plt.ylabel("normalized median score", fontsize=10)
plt.legend(fontsize=8, bbox_to_anchor=(1.05, 1), loc="upper left",␣
 ↪borderaxespad=0.0)
plt.grid()

plt.figure(figsize=(5, 5), dpi=300)
for method_ver_name in summary_ds.coords[METHOD].values:
    curr_ds = summary_ds.sel({METHOD: method_ver_name})
    curr_ds.coords[ITER].values

    plt.fill_between(
        curr_ds.coords[ITER].values,
        curr_ds[cc.LB_MEAN].values,
        curr_ds[cc.UB_MEAN].values,
        color=method_to_rgba[method_ver_name],
        alpha=0.5,
    )
    plt.plot(
        curr_ds.coords[ITER].values,
        curr_ds[cc.PERF_MEAN].values,
        color=method_to_rgba[method_ver_name],
        label=method_ver_name,
        marker=".",
    )
plt.xlabel("evaluation", fontsize=10)
plt.ylabel("mean score", fontsize=10)
plt.legend(fontsize=8, bbox_to_anchor=(1.05, 1), loc="upper left",␣
 ↪borderaxespad=0.0)
plt.grid()

plt.figure(figsize=(5, 5), dpi=300)
for method_ver_name in summary_ds.coords[METHOD].values:
    curr_ds = summary_ds.sel({METHOD: method_ver_name})
    curr_ds.coords[ITER].values

    plt.fill_between(
        curr_ds.coords[ITER].values,
        curr_ds[cc.LB_NORMED_MEAN].values,
        curr_ds[cc.UB_NORMED_MEAN].values,
        color=method_to_rgba[method_ver_name],
        alpha=0.5,
    )
    plt.plot(
        curr_ds.coords[ITER].values,
        curr_ds[cc.NORMED_MEAN].values,
        color=method_to_rgba[method_ver_name],
```

```
        label=method_ver_name,
        marker=".",
    )
plt.xlabel("evaluation", fontsize=10)
plt.ylabel("normalized mean score", fontsize=10)
plt.legend(fontsize=8, bbox_to_anchor=(1.05, 1), loc="upper left",␣
 ↪borderaxespad=0.0)
plt.grid()
```




mean score
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20
40
60
80
100
evaluation
HyperOpt_0.1.1_5cf1fc7
Nevergrad-OnePlusOne_0.1.4_5cf1fc7
OpenTuner-BanditA_0.8.1_5cf1fc7
OpenTuner-GA-DE_0.8.1_5cf1fc7
OpenTuner-GA_0.8.1_5cf1fc7
PySOT_0.2.3_5cf1fc7
RandomSearch_0.0.4_5cf1fc7
Scikit-GBRT-Hedge_0.5.2_5cf1fc7
Scikit-GP-Hedge_0.5.2_5cf1fc7
Scikit-GP-LCB_0.5.2_5cf1fc7


normalized mean score
1.2
1.0
0.8
0.6
0.4
0
20
40
60
80
100
evaluation
HyperOpt_0.1.1_5cf1fc7
Nevergrad-OnePlusOne_0.1.4_5cf1fc7
OpenTuner-BanditA_0.8.1_5cf1fc7
OpenTuner-GA-DE_0.8.1_5cf1fc7
OpenTuner-GA_0.8.1_5cf1fc7
PySOT_0.2.3_5cf1fc7
RandomSearch_0.0.4_5cf1fc7
Scikit-GBRT-Hedge_0.5.2_5cf1fc7
Scikit-GP-Hedge_0.5.2_5cf1fc7
Scikit-GP-LCB_0.5.2_5cf1fc7