In [1]:
```
import sys, pandas as pd, openpyxl
print(sys.executable)              # should point to ...\anaconda3\envs\asu\python.
print("pandas:", pd.__version__, "openpyxl:", openpyxl.__version__)
```

```
C:\Users\jayso\anaconda3\envs\asu\python.exe
pandas: 2.3.2 openpyxl: 3.1.5
```

In [2]:
```
from pathlib import Path
import pandas as pd

candidate = Path(r"C:\Users\jayso\Downloads\Teamwork-regress w 2 losses (dataset)-1
df = pd.read_excel(candidate, engine="openpyxl")
df.head()
```

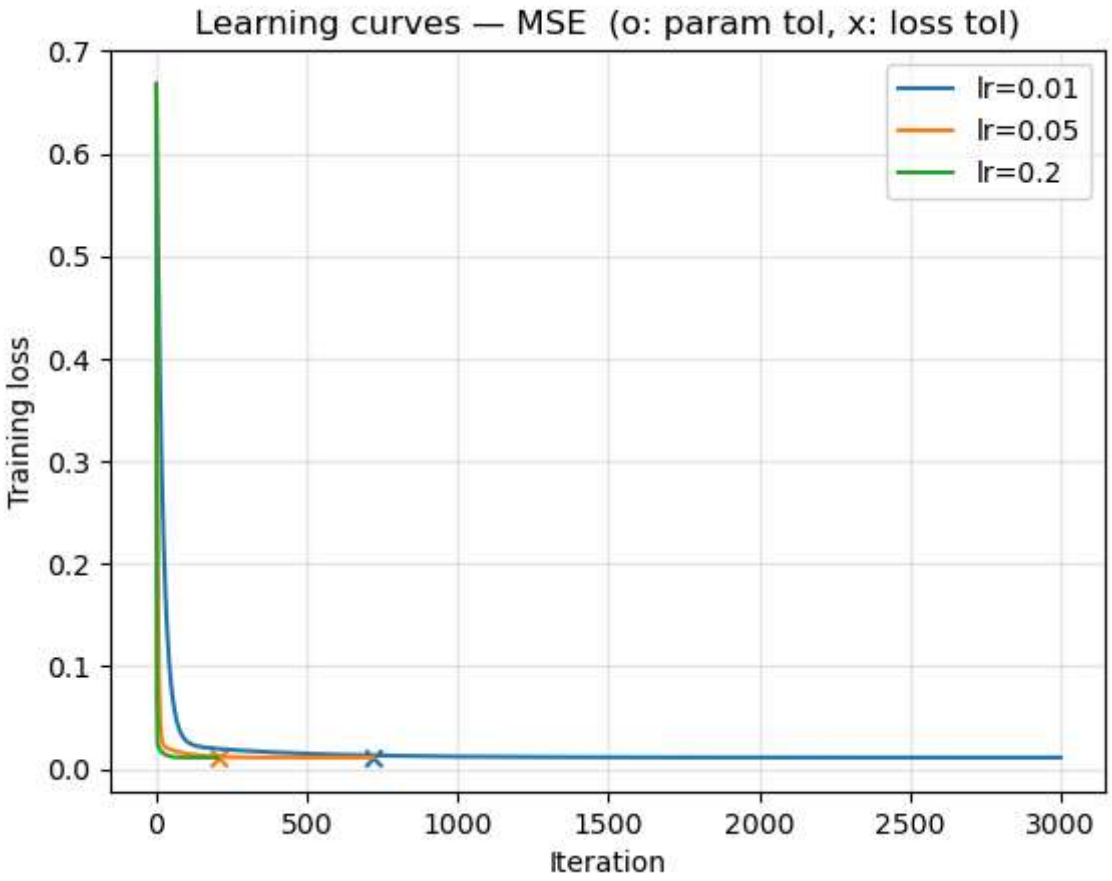
Out[2]:

| | x | y |
|---|---|---|
| **0** | 0.417411 | 0.841049 |
| **1** | 0.222108 | 0.556829 |
| **2** | 0.119865 | 0.518283 |
| **3** | 0.337615 | 0.788053 |
| **4** | 0.942910 | 1.067603 |

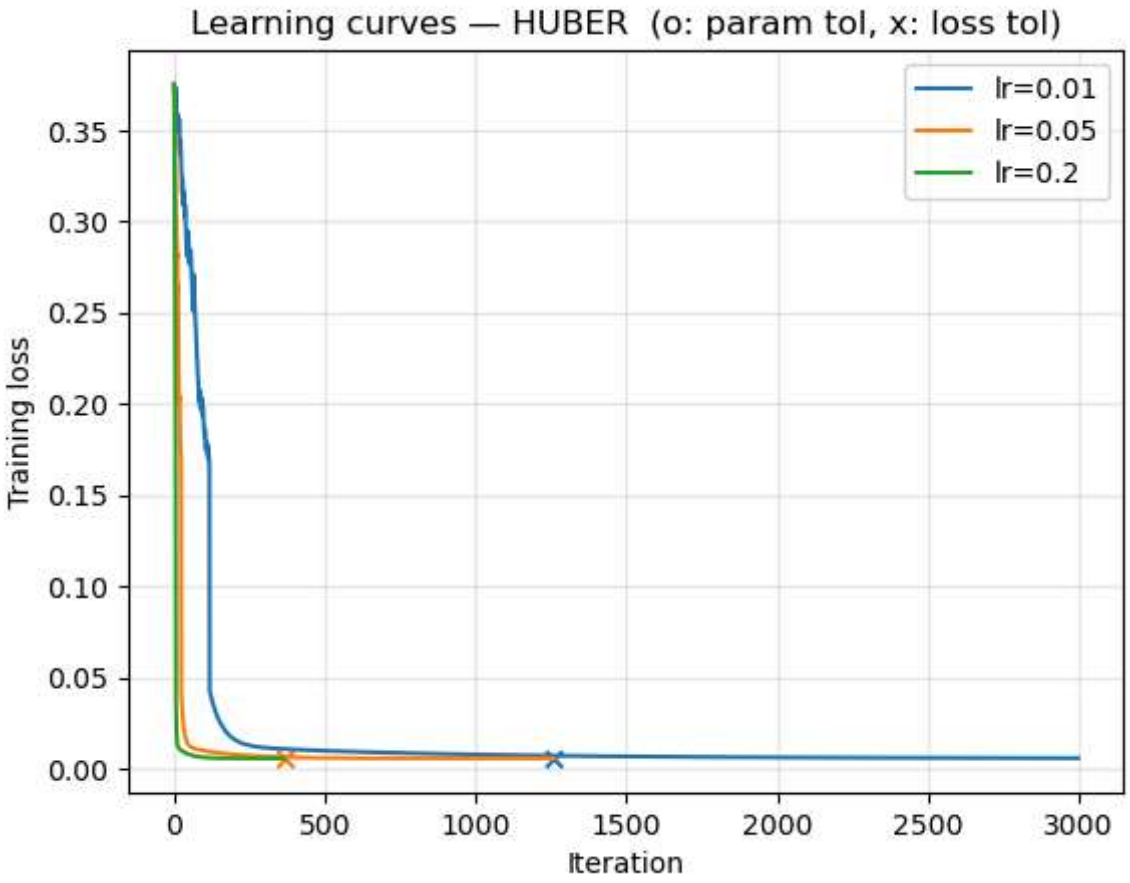
In [6]:
```
# === RUN EXPERIMENT ===
import numpy as np, matplotlib.pyplot as plt

# Design matrix for y = w*x + b
X = np.c_[df["x"].to_numpy(), np.ones(len(df))]
y = df["y"].to_numpy().reshape(-1,1)

lrs   = [0.01, 0.05, 0.2]
delta = 0.5

for loss in ["mse", "huber"]:
    print(f"\n=== {loss.upper()} ===")
    plt.figure()
    for lr in lrs:
        theta, losses, param_deltas, k_param, k_loss = gd_linear_regression(
            X, y, loss=loss, lr=lr, delta=delta, max_iters=3000,
            tol_param=1e-6, tol_loss=1e-8
        )
        print(f"lr={lr:<4}  theta=[{theta[0,0]:.4f}, {theta[1,0]:.4f}]  "
              f"final_loss={losses[-1]:.6f}  "
              f"stop(param)={k_param}  stop(loss)={k_loss}  iters={len(losses)}")

        # plot curve + markers for stop points
        it = np.arange(len(losses))
        plt.plot(it, losses, label=f"lr={lr}")
        if k_param is not None and k_param < len(losses):
            plt.scatter([k_param], [losses[k_param]], marker='o')   # param tol met
        if k_loss is not None and k_loss < len(losses):
            plt.scatter([k_loss], [losses[k_loss]], marker='x')     # loss tol met
```

```
    plt.xlabel("Iteration"); plt.ylabel("Training loss")
    plt.title(f"Learning curves — {loss.upper()}  (o: param tol, x: loss tol)")
    plt.legend(); plt.grid(True, alpha=0.3)
    plt.show()
```

```
=== MSE ===
lr=0.01  theta=[0.7266, 0.4607]  final_loss=0.011105  stop(param)=None  stop(loss)=N
one  iters=3000
lr=0.05  theta=[0.7305, 0.4589]  final_loss=0.011102  stop(param)=None  stop(loss)=7
21  iters=722
lr=0.2   theta=[0.7320, 0.4581]  final_loss=0.011102  stop(param)=None  stop(loss)=2
04  iters=205
```




```
=== HUBER ===
lr=0.01  theta=[0.6768, 0.4844]  final_loss=0.005686  stop(param)=None  stop(loss)=N
one  iters=3000
lr=0.05  theta=[0.7276, 0.4603]  final_loss=0.005552  stop(param)=None  stop(loss)=1
259  iters=1260
lr=0.2   theta=[0.7306, 0.4588]  final_loss=0.005551  stop(param)=None  stop(loss)=3
65  iters=366
```




In [7]:
```
import numpy as np

theta_ls, *_ = np.linalg.lstsq(X, y, rcond=None)
print("LS  theta:", theta_ls.ravel())

theta_mse, *_ = gd_linear_regression(X, y, loss="mse", lr=0.05, max_iters=3000)
print("GD  theta:", theta_mse.ravel())

print("||GD - LS||:", np.linalg.norm(theta_mse - theta_ls))
```

```
LS  theta: [0.73340398 0.45748571]
GD  theta: [0.73049922 0.45886703]
||GD - LS||: 0.003216470784084773
```

In [ ]: