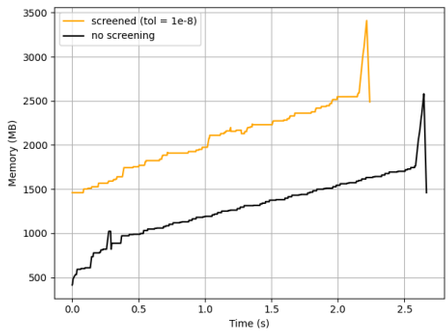
In [1]:
```python
### loading
import numpy as np
import matplotlib.pyplot as plt
import time
import threading
import psutil
import os
from iodata import load_one

from grid.becke import BeckeWeights
from grid.molgrid import MolGrid
from grid.onedgrid import GaussChebyshev
from grid.rtransform import BeckeRTransform

from gbasis.wrappers import from_iodata
from gbasis import parsers

## iron hexacarbonyl
iron = load_one("/Users/omid/Downloads/iron_hexacarbonyl.fchk")
iron_rdm = iron.one_rdms['scf']
iron_basis = from_iodata(iron)

becke = BeckeWeights(order=3)
aim_weights = BeckeWeights(radii=None, order=3)
oned = GaussChebyshev(100)
rgrid = BeckeRTransform(1e-4, 1.5).transform_1d_grid(oned)
grid = MolGrid.from_preset(iron.atnums, iron.atcoords, 'insane', rgrid, aim_weights=aim_weights)
iron_points = grid.points

## a pentapeptide, 62 atoms
pep = load_one("/Users/omid/Downloads/0002_q000_m01_k00_force_uhf_def2svpd.fchk")
pep_rdm = pep.one_rdms['scf']
pep_basis = from_iodata(pep)

coord_limit = 10
plt_pts = 500
plt_lmts = [-coord_limit, coord_limit, -coord_limit, coord_limit]
x_vals = np.linspace(-coord_limit, coord_limit, plt_pts)
y_vals = x_vals
x_vals, y_vals = np.meshgrid(x_vals, y_vals)
x_vals, y_vals = x_vals.flatten(), y_vals.flatten()
z_vals = np.zeros_like(x_vals)
pep_points = np.array([x_vals, y_vals, z_vals]).T

# ram monitoring
def monitor_memory(stop_event, timestamps, mem_usage):
    process = psutil.Process(os.getpid())
    while not stop_event.is_set():
        timestamps.append(time.perf_counter())
        mem_usage.append(process.memory_info().rss / (1024 * 1024))
        time.sleep(0.001)

def run_with_monitoring(func):
    timestamps = []
    mem_usage = []
    stop_event = threading.Event()
    monitor_thread = threading.Thread(
        target=monitor_memory, args=(stop_event, timestamps, mem_usage)
    )

    start = time.perf_counter()
    monitor_thread.start()

    result = func()

    stop_event.set()
    monitor_thread.join()
    end = time.perf_counter()

    timestamps = [t - start for t in timestamps]

    return result, timestamps, mem_usage

## orders for derivatives
orders = np.array([1, 1, 1])
```

```
/Users/omid/GSoC/gbasis/env/lib/python3.13/site-packages/grid/atomgrid.py:896: UserWarning: Lebedev weights are negative which can introduce round-off errors.
  sphere_grid = AngularGrid(degree=deg_i, method=method)
```

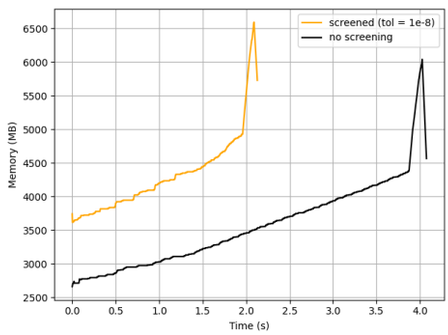
In [2]:
```python
## iron basis set evaluation
from gbasis.evals.eval import evaluate_basis

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_basis(iron_basis, iron_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_basis(iron_basis, iron_points))

print(f"Are screened and unscreened basis evaluations identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened basis evaluations identical within 1e-4? True
```



In [3]:
```python
## peptide basis set evaluation
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_basis(pep_basis, pep_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_basis(pep_basis, pep_points))

print(f"Are screened and unscreened basis evaluations identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

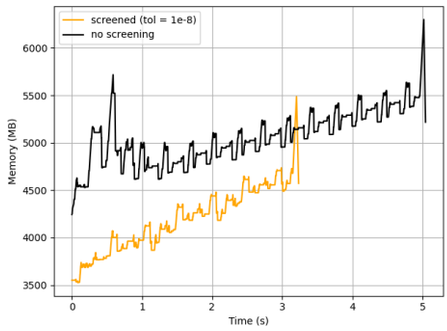
```
Are screened and unscreened basis evaluations identical within 1e-4? True
```



In [4]:
```python
## iron basis derivative evaluations
from gbasis.evals.eval_deriv import evaluate_deriv_basis

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_deriv_basis(iron_basis, iron_points, orders, deriv_type="direct", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_deriv_basis(iron_basis, iron_points, orders, deriv_type="direct"))
```

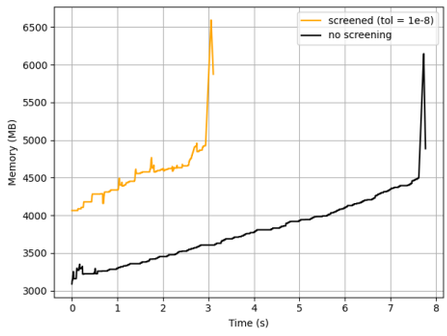
```python
print(f"Are screened and unscreened basis set derivatives identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened basis set derivatives identical within 1e-4? False
```



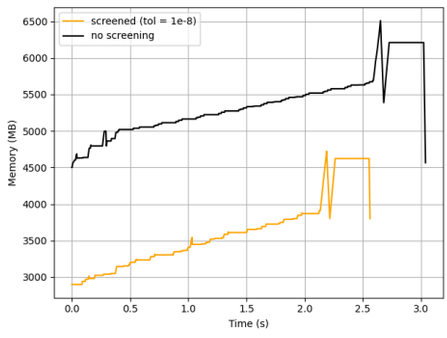
In [5]:
```python
## peptide basis derivative evaluations
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_deriv_basis(pep_basis, pep_points, orders, deriv_type="direct", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_deriv_basis(pep_basis, pep_points, orders, deriv_type="direct"))

print(f"Are screened and unscreened basis set derivatives identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened basis set derivatives identical within 1e-4? False
```



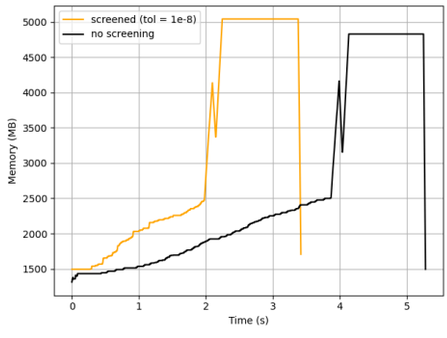
In [6]:
```python
## iron density evaluation
from gbasis.evals.density import evaluate_density

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_density(iron_rdm, iron_basis, iron_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_density(iron_rdm, iron_basis, iron_points))

print(f"Are screened and unscreened density evaluations identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened density evaluations identical within 1e-4? True
```



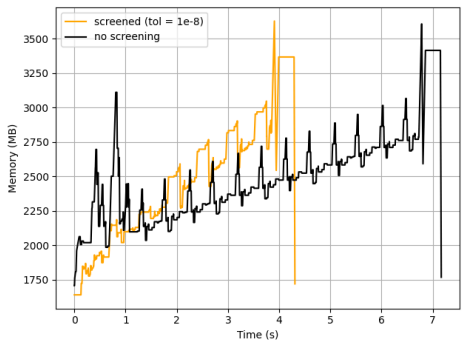
In [7]:
```python
##peptide density eavluation
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_density(pep_rdm, pep_basis, pep_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_density(pep_rdm, pep_basis, pep_points))

print(f"Are screened and unscreened density evaluations identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened density evaluations identical within 1e-4? True
```



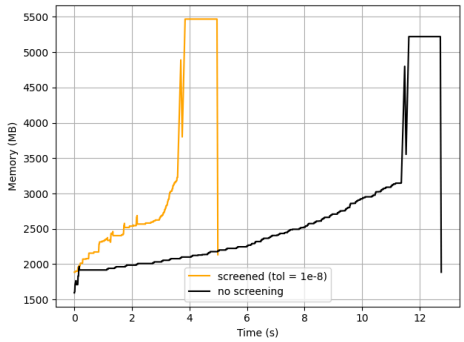
In [8]:
```python
## iron rdm derivative evaluations
from gbasis.evals.density import evaluate_deriv_reduced_density_matrix

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_deriv_reduced_density_matrix(orders, orders, iron_rdm, iron_basis, iron_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_deriv_reduced_density_matrix(orders, orders, iron_rdm, iron_basis, iron_points, deriv_type="general"))

print(f"Are screened and unscreened RDM derivatives identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened RDM derivatives identical within 1e-4? False
```



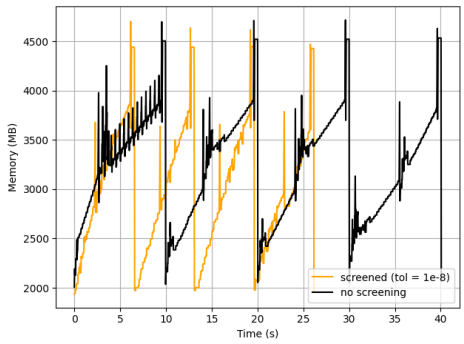
In [9]:
```python
## peptide rdm derivative evaluations
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_deriv_reduced_density_matrix(orders, orders, pep_rdm, pep_basis, pep_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_deriv_reduced_density_matrix(orders, orders, pep_rdm, pep_basis, pep_points, deriv_type="general"))

print(f"Are screened and unscreened RDM derivatives identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened RDM derivatives identical within 1e-4? True
```



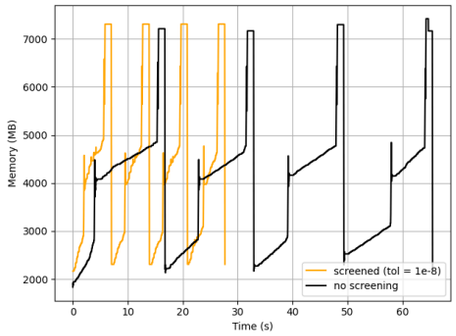
In [10]:
```python
## iron density derivative evaluations
from gbasis.evals.density import evaluate_deriv_density

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_deriv_density(orders, iron_rdm, iron_basis, iron_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_deriv_density(orders, iron_rdm, iron_basis, iron_points, deriv_type="general"))

print(f"Are screened and unscreened density derivatives identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened density derivatives identical within 1e-4? False
```



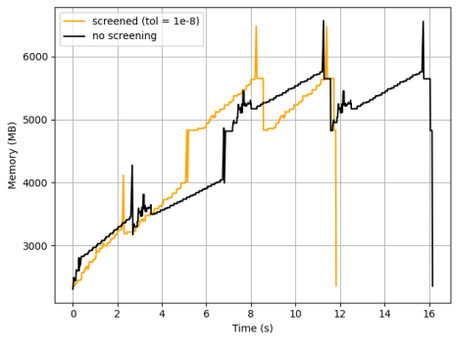
In [11]:
```python
## peptide density derivative evaluations
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_deriv_density(orders, pep_rdm, pep_basis, pep_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_deriv_density(orders, pep_rdm, pep_basis, pep_points, deriv_type="general"))

print(f"Are screened and unscreened density derivatives identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened density derivatives identical within 1e-4? True
```



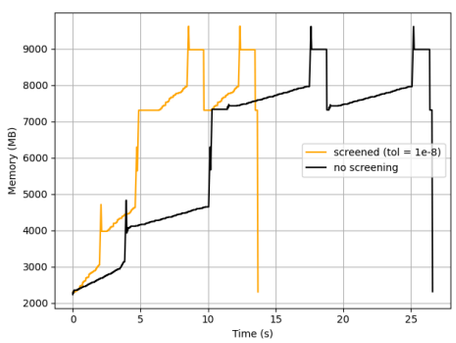
In [12]:
```
## iron density gradient evaluations
from gbasis.evals.density import evaluate_density_gradient

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_density_gradient(iron_rdm, iron_basis, iron_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_density_gradient(iron_rdm, iron_basis, iron_points, deriv_type="general"))

print(f"Are screened and unscreened density gradient identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened density gradient identical within 1e-4? True
```



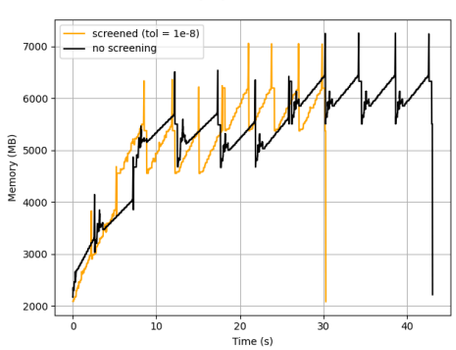
In [13]:
```
## peptide density gradient evaluations
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_density_gradient(pep_rdm, pep_basis, pep_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_density_gradient(pep_rdm, pep_basis, pep_points, deriv_type="general"))

print(f"Are screened and unscreened basis set derivatives identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened basis set derivatives identical within 1e-4? True
```



In [14]:
```
## iron density laplacian evaluations
from gbasis.evals.density import evaluate_density_laplacian

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_density_laplacian(iron_rdm, iron_basis, iron_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_density_laplacian(iron_rdm, iron_basis, iron_points, deriv_type="general"))

print(f"Are screened and unscreened density laplacian identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

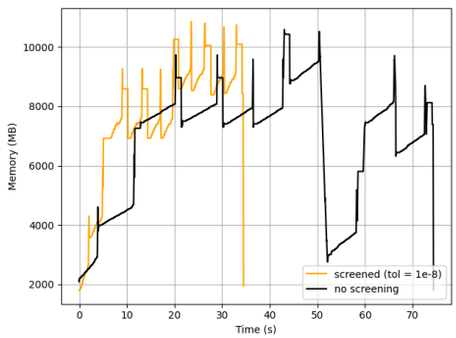
```
Are screened and unscreened density laplacian identical within 1e-4? False
```



In [15]:
```
## peptide density laplacian evaluations
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_density_laplacian(pep_rdm, pep_basis, pep_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_density_laplacian(pep_rdm, pep_basis, pep_points, deriv_type="general"))

print(f"Are screened and unscreened density laplacian identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")
```

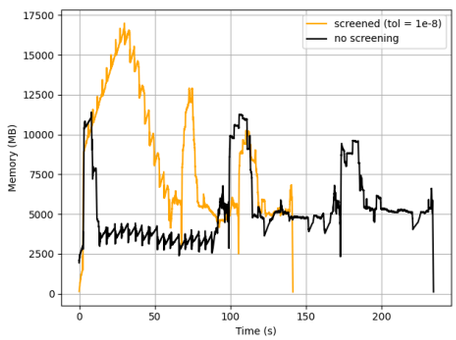
```
plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened density laplacian identical within 1e-4? True
```



In [16]:
```
## iron density hessian evaluations
from gbasis.evals.density import evaluate_density_hessian

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_density_hessian(iron_rdm, iron_basis, iron_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_density_hessian(iron_rdm, iron_basis, iron_points, deriv_type="general"))

print(f"Are screened and unscreened density hessian identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

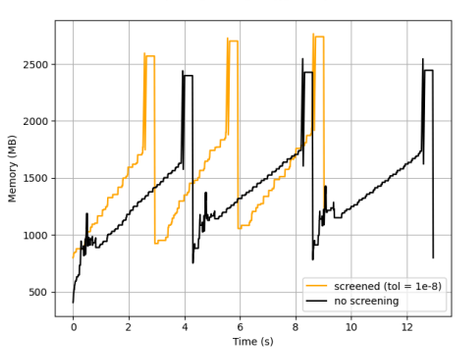
```
Are screened and unscreened density hessian identical within 1e-4? False
```



In [ ]:
```
## peptide density hessian evaluations
from gbasis.evals.density import evaluate_density_hessian

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_density_hessian(pep_rdm, pep_basis, pep_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_density_hessian(pep_rdm, pep_basis, pep_points, deriv_type="general"))

print(f"Are screened and unscreened density hessian identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()

## I wasn't able to complete this run, the kernel kept dying in the middle of the run
```

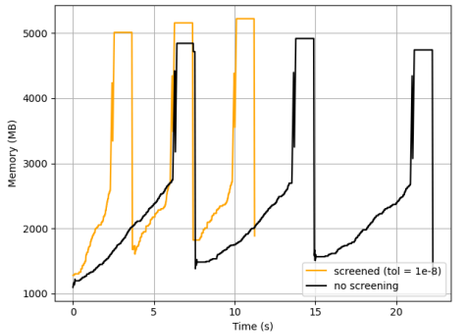
In [2]:
```
## iron posdef_kinetic_energy_density evaluations
from gbasis.evals.density import evaluate_posdef_kinetic_energy_density

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_posdef_kinetic_energy_density(iron_rdm, iron_basis, iron_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_posdef_kinetic_energy_density(iron_rdm, iron_basis, iron_points, deriv_type="general"))

print(f"Are screened and unscreened posdef_kinetic_energy_density identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened posdef_kinetic_energy_density identical within 1e-4? True
```



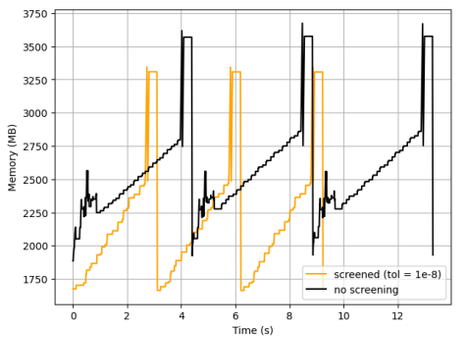
In [3]:
```
## peptide posdef_kinetic_energy_density evaluations
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_posdef_kinetic_energy_density(pep_rdm, pep_basis, pep_points, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_posdef_kinetic_energy_density(pep_rdm, pep_basis, pep_points, deriv_type="general"))

print(f"Are screened and unscreened posdef_kinetic_energy_density identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened posdef_kinetic_energy_density identical within 1e-4? True
```



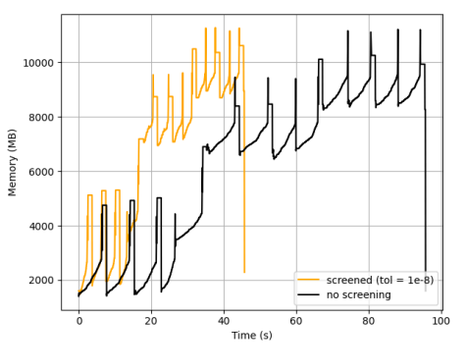
In [4]:
```
## iron general_kinetic_energy_density evaluations
from gbasis.evals.density import evaluate_general_kinetic_energy_density

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_general_kinetic_energy_density(iron_rdm, iron_basis, iron_points, alpha=0, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_general_kinetic_energy_density(iron_rdm, iron_basis, iron_points, alpha=0, deriv_type="general"))

print(f"Are screened and unscreened general_kinetic_energy_density identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened general_kinetic_energy_density identical within 1e-4? True
```



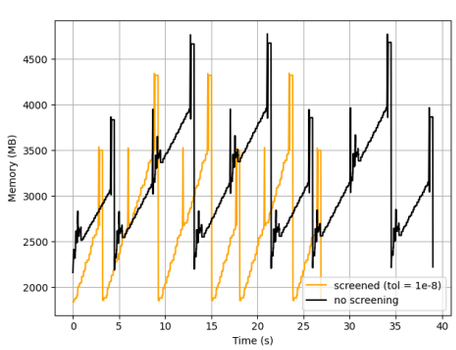
In [5]:
```
## peptide general_kinetic_energy_density evaluations
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_general_kinetic_energy_density(pep_rdm, pep_basis, pep_points, alpha=1, deriv_type="general", screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_general_kinetic_energy_density(pep_rdm, pep_basis, pep_points, alpha=1, deriv_type="general"))

print(f"Are screened and unscreened general_kinetic_energy_density identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened general_kinetic_energy_density identical within 1e-4? True
```



In [6]:
```
## iron stress tensor evaluations
from gbasis.evals.stress_tensor import evaluate_stress_tensor

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_stress_tensor(iron_rdm, iron_basis, iron_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_stress_tensor(iron_rdm, iron_basis, iron_points))

print(f"Are screened and unscreened stress tensors identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

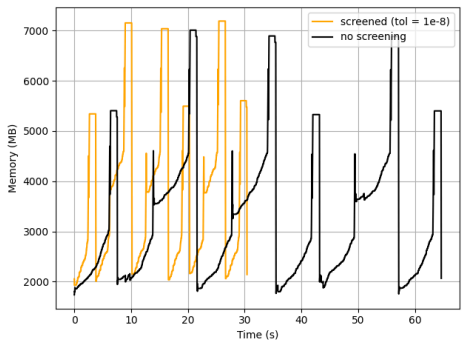
```
Are screened and unscreened stress tensors identical within 1e-4? True
```



In [7]:
```
## peptide stress tensor evaluations
no_screen, tn, mn = run_with_monitoring(lambda: evaluate_stress_tensor(pep_rdm, pep_basis, pep_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_stress_tensor(pep_rdm, pep_basis, pep_points))

print(f"Are screened and unscreened stress tensors identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")
```

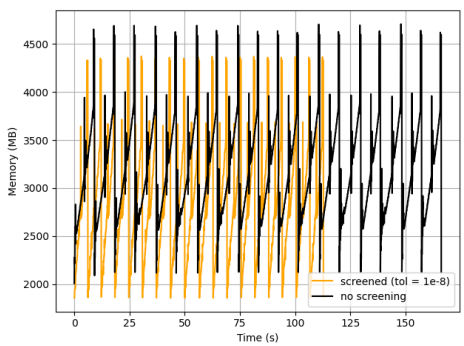
```
plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened stress tensors identical within 1e-4? True
```



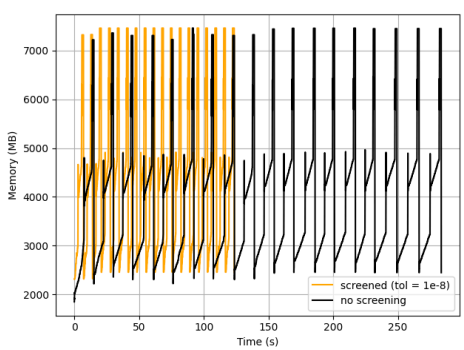
In [8]:
```
## iron ehrenfest force evaluations
from gbasis.evals.stress_tensor import evaluate_ehrenfest_force

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_ehrenfest_force(iron_rdm, iron_basis, iron_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_ehrenfest_force(iron_rdm, iron_basis, iron_points))

print(f"Are screened and unscreened ehrenfest forces identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened ehrenfest forces identical within 1e-4? False
```



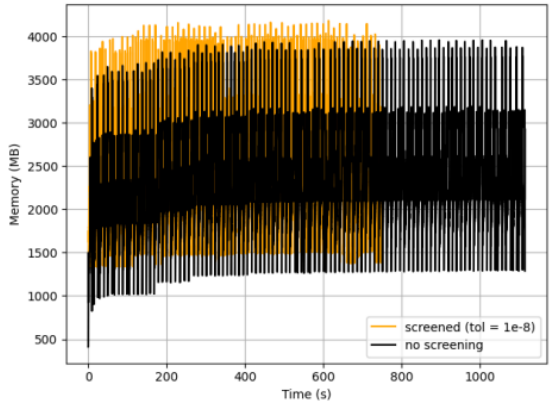
In [9]:
```
## peptide ehrenfest force evaluations
from gbasis.evals.stress_tensor import evaluate_ehrenfest_force

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_ehrenfest_force(pep_rdm, pep_basis, pep_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_ehrenfest_force(pep_rdm, pep_basis, pep_points))

print(f"Are screened and unscreened ehrenfest forces identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Are screened and unscreened ehrenfest forces identical within 1e-4? True
```



In [2]:
```
## iron ehrenfest hessians evaluations
from gbasis.evals.stress_tensor import evaluate_ehrenfest_hessian

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_ehrenfest_hessian(iron_rdm, iron_basis, iron_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_ehrenfest_hessian(iron_rdm, iron_basis, iron_points))

print(f"Are screened and unscreened ehrenfest hessians identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()

## in this case, the integrals are really not equal, even with atol=0.1
```

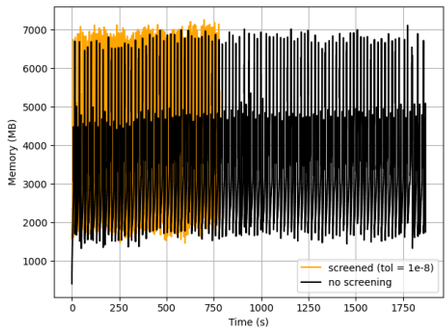
```
Are screened and unscreened ehrenfest hessians identical within 1e-4? False
```



In [3]:

```
## peptide ehrenfest hessians evaluations
from gbasis.evals.stress_tensor import evaluate_ehrenfest_hessian

no_screen, tn, mn = run_with_monitoring(lambda: evaluate_ehrenfest_hessian(pep_rdm, pep_basis, pep_points, screen_basis=False))
screen, ts, ms = run_with_monitoring(lambda: evaluate_ehrenfest_hessian(pep_rdm, pep_basis, pep_points))

print(f"Are screened and unscreened ehrenfest hessians identical within 1e-4? {np.allclose(no_screen, screen, atol=1e-4)} \n")

plt.figure()
plt.plot(ts, ms, label="screened (tol = 1e-8)", color='orange')
plt.plot(tn, mn, label="no screening", color='black')
plt.xlabel("Time (s)")
plt.ylabel("Memory (MB)")
plt.title("")
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()

## in this case, the integrals are equal with atol=0.3
```

```
Are screened and unscreened ehrenfest hessians identical within 1e-4? False
```


7000
6000
5000
4000
3000
2000
1000
Memory (MB)
0
250
500
750
1000
1250
1500
1750
Time (s)
screened (tol = 1e-8)
no screening