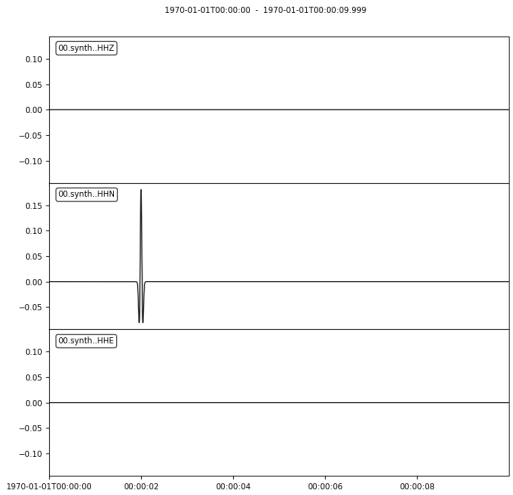
# Simple synthetic example

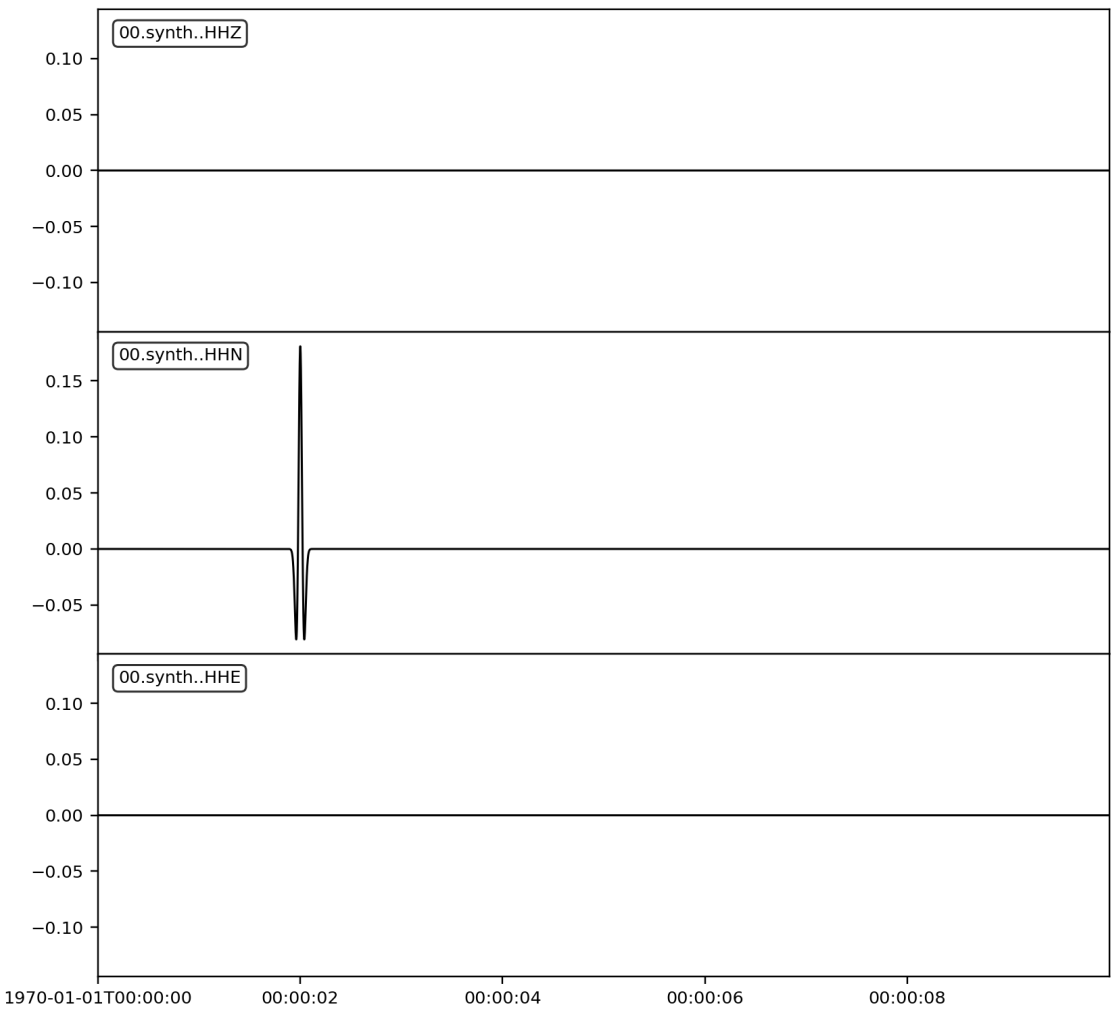
In [1]:
```
%load_ext autoreload
%autoreload 2
```

In [2]:
```
import swspy
import numpy as np
import os, sys
import matplotlib
import matplotlib.pyplot as plt
from scipy import signal
%matplotlib notebook
```

## 1. Create source-time function:

In [13]:
```
# Create source-time function:
t_src = 2.0
src_pol_from_N = 0.0
ZNE_st = swspy.splitting.forward_model.create_src_time_func(10., 1000, src_po
ZNE_st.plot()
```





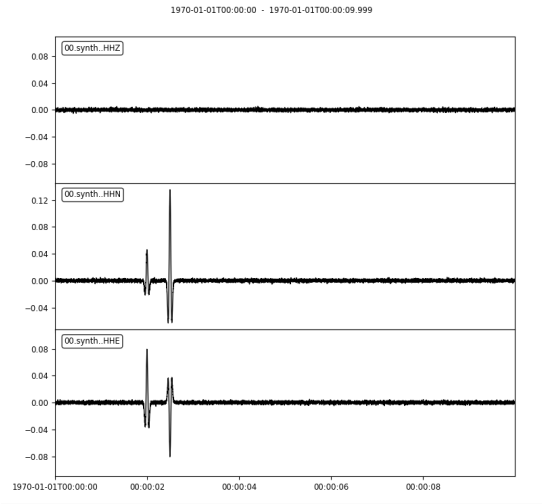

## 2. Apply a layer of splitting:

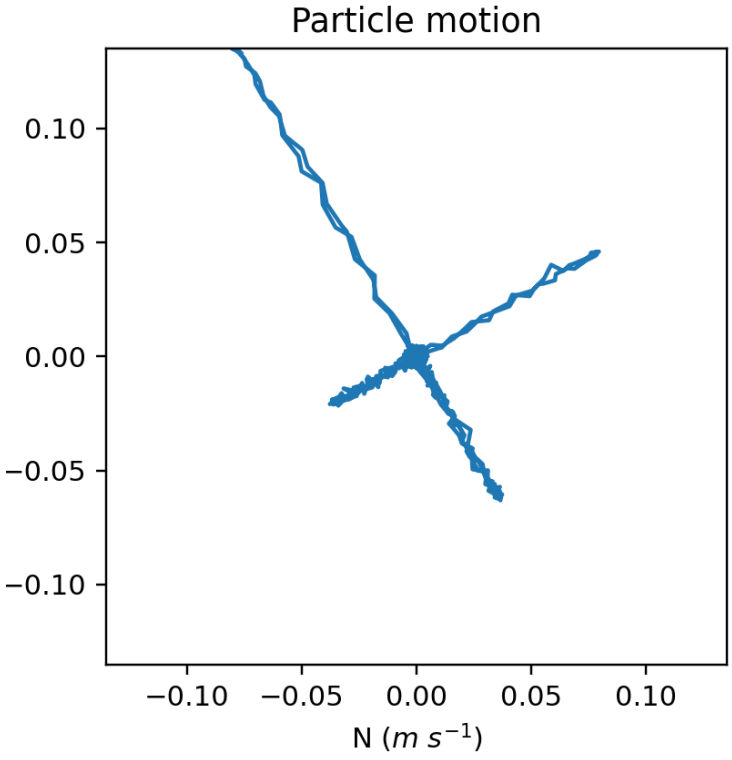
In [14]:
```
# Define SNR:
snr = 100 #1000 #100 # SNR of the src time func (applied so that get non-zero
```

In [15]:
```
# Specify layer anisotropy parameters:
phi_from_N = 60.
dt = 0.5 #0.05
back_azi = 0
event_inclin_angle_at_station = 0

# Apply splitting:
ZNE_st_layer1 = swspy.splitting.forward_model.add_splitting(ZNE_st, phi_from_
ZNE_st_layer1.plot()

plt.figure(figsize=(4,4))
plt.plot(ZNE_st_layer1.select(channel="??E")[0].data, ZNE_st_layer1.select(ch
abs_max_tmp = np.max(np.array([np.max(np.abs(ZNE_st_layer1.select(channel="??
plt.xlim([-abs_max_tmp, abs_max_tmp])
plt.ylim([-abs_max_tmp, abs_max_tmp])
plt.title("Particle motion")
plt.xlabel("E ($m$ $s^{-1}$)")
plt.xlabel("N ($m$ $s^{-1}$)")
plt.show()
```



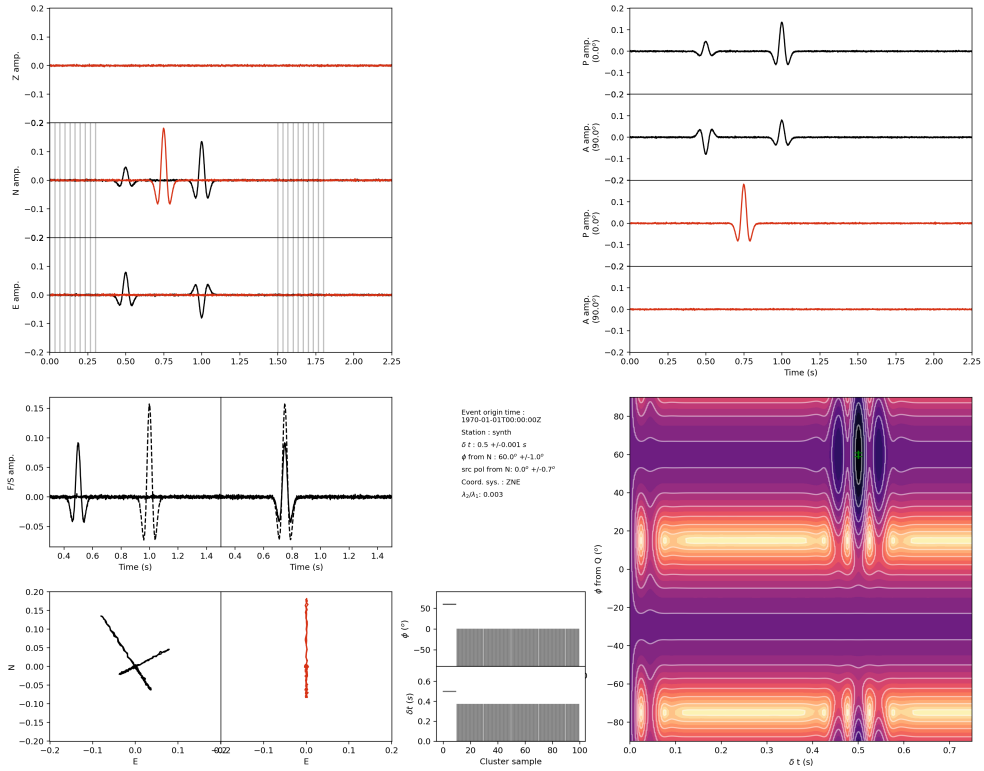




## 3. Measure splitting on single layer:

In [16]:
```
# Measure splitting:
event_uid = "single-layer"
S_phase_arrival_times = [ZNE_st_layer1[0].stats.starttime+t_src]
back_azis_all_stations = [back_azi]
receiver_inc_angles_all_stations = [event_inclin_angle_at_station]
splitting_event = swspy.splitting.create_splitting_object(ZNE_st_layer1, even
splitting_event.overall_win_start_pre_fast_S_pick = 0.5
splitting_event.win_S_pick_tolerance = 0.2
splitting_event.overall_win_start_post_fast_S_pick = 1.0
splitting_event.rotate_step_deg = 2 #1.0
splitting_event.max_t_shift_s = 0.75 #1.0
splitting_event.n_win = 10
splitting_event.perform_sws_analysis(coord_system="ZNE", sws_method="EV")

# And plot splitting result:
splitting_event.plot(outdir=os.path.join("outputs", "plots"))

# And save result to file:
splitting_event.save_result(outdir=os.path.join("outputs", "data"))
```




```
/Users/eart0504/Documents/python/github_repositories/swspy/swspy/splitting/spl
it.py:1655: UserWarning: constrained_layout not applied.  At least one axes co
llapsed to zero width or height.
  plt.savefig(os.path.join(outdir, ''.join((self.event_uid, "_", station, ".pn
g"))), dpi=300)
Saved sws result to: outputs/data/single-layer_sws_result.csv
```

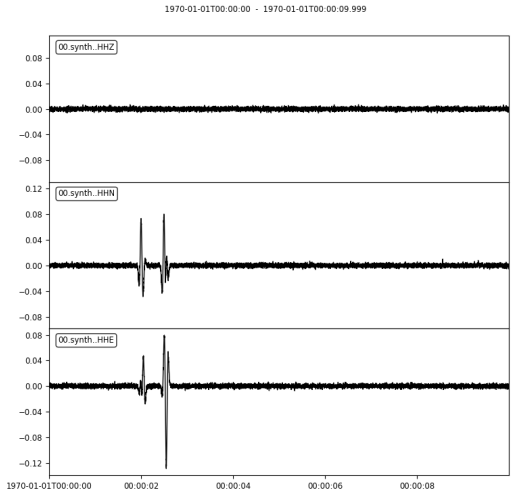
# Multi-layer splitting

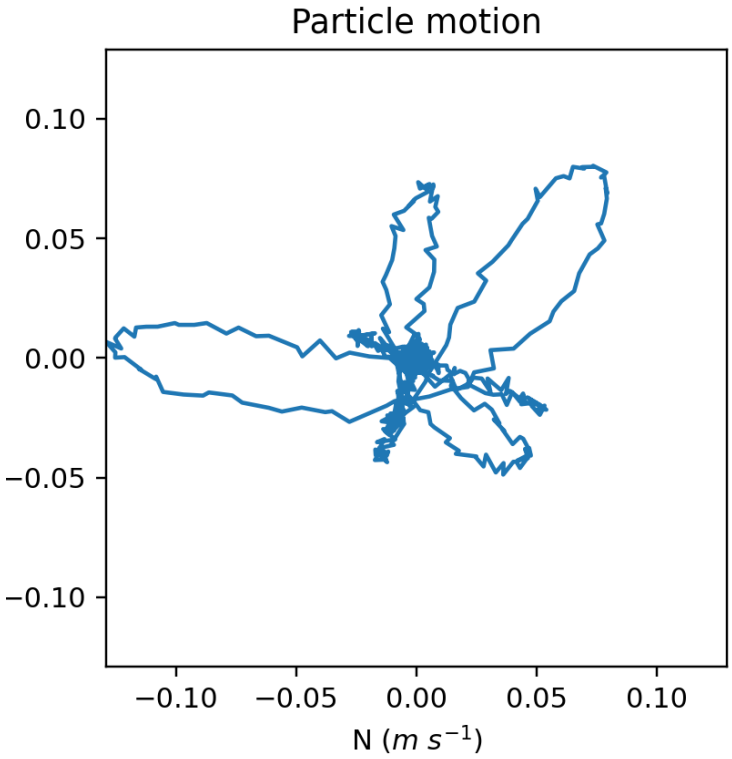
## 4. Apply a second layer of splitting:

In [17]:
```
# Specify second layer parameters:
phi_from_N = 20.
dt = 0.05
# back_azi = 0
# event_inclin_angle_at_station = 0

# Apply splitting:
ZNE_st_layer1and2 = swspy.splitting.forward_model.add_splitting(ZNE_st_layer1
ZNE_st_layer1and2.plot()

plt.figure(figsize=(4,4))
plt.plot(ZNE_st_layer1and2.select(channel="??E")[0].data, ZNE_st_layer1and2.s
abs_max_tmp = np.max(np.array([np.max(np.abs(ZNE_st_layer1and2.select(channel
plt.xlim([-abs_max_tmp, abs_max_tmp])
plt.ylim([-abs_max_tmp, abs_max_tmp])
plt.title("Particle motion")
plt.xlabel("E ($m$ $s^{-1}$)")
plt.xlabel("N ($m$ $s^{-1}$)")
plt.show()
```



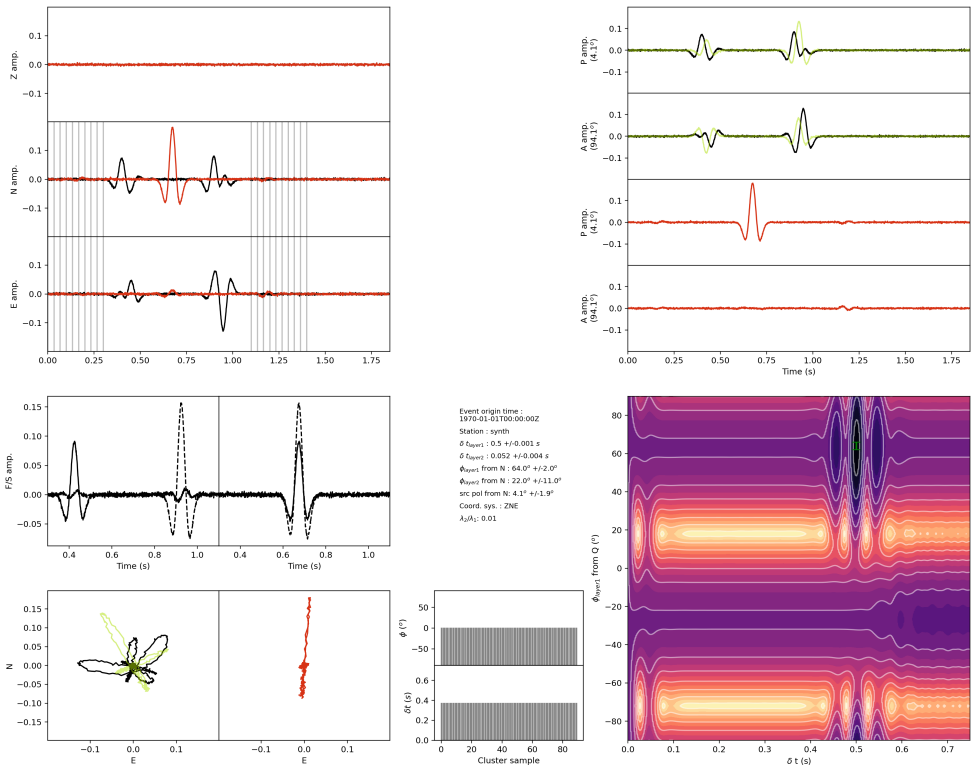



## 5. And measure multi-layer splitting:

In [18]:
```
# Measure splitting for multi-layer:
event_uid = "multi-layer"
S_phase_arrival_times = [ZNE_st_layer1and2[0].stats.starttime+t_src]
back_azis_all_stations = [back_azi]
receiver_inc_angles_all_stations = [event_inclin_angle_at_station]
splitting_event_multi_layer = swspy.splitting.create_splitting_object(ZNE_st_
splitting_event_multi_layer.overall_win_start_pre_fast_S_pick = 0.4 #0.3 #0.5
splitting_event_multi_layer.win_S_pick_tolerance = 0.1
splitting_event_multi_layer.overall_win_start_post_fast_S_pick = 0.7 #0.8 #1.
splitting_event_multi_layer.rotate_step_deg = 2 #1. #2.5 #2. #1. #5 #2.0
splitting_event_multi_layer.max_t_shift_s = 0.75 #0.7 #1.0
splitting_event_multi_layer.n_win = 10
splitting_event_multi_layer.perform_sws_analysis_multi_layer(coord_system="ZN

# Plot and save result:
splitting_event_multi_layer.plot(outdir=os.path.join("outputs", "plots"))
splitting_event_multi_layer.save_result(outdir=os.path.join("outputs", "data"
```

```
Passed multi-layer result, therefore plotting this result.
```




```
/Users/eart0504/Documents/python/github_repositories/swspy/swspy/splitting/spl
it.py:1655: UserWarning: constrained_layout not applied.  At least one axes co
llapsed to zero width or height.
  plt.savefig(os.path.join(outdir, ''.join((self.event_uid, "_", station, ".pn
g"))), dpi=300)
Saved sws result to: outputs/data/multi-layer_sws_result.csv
```

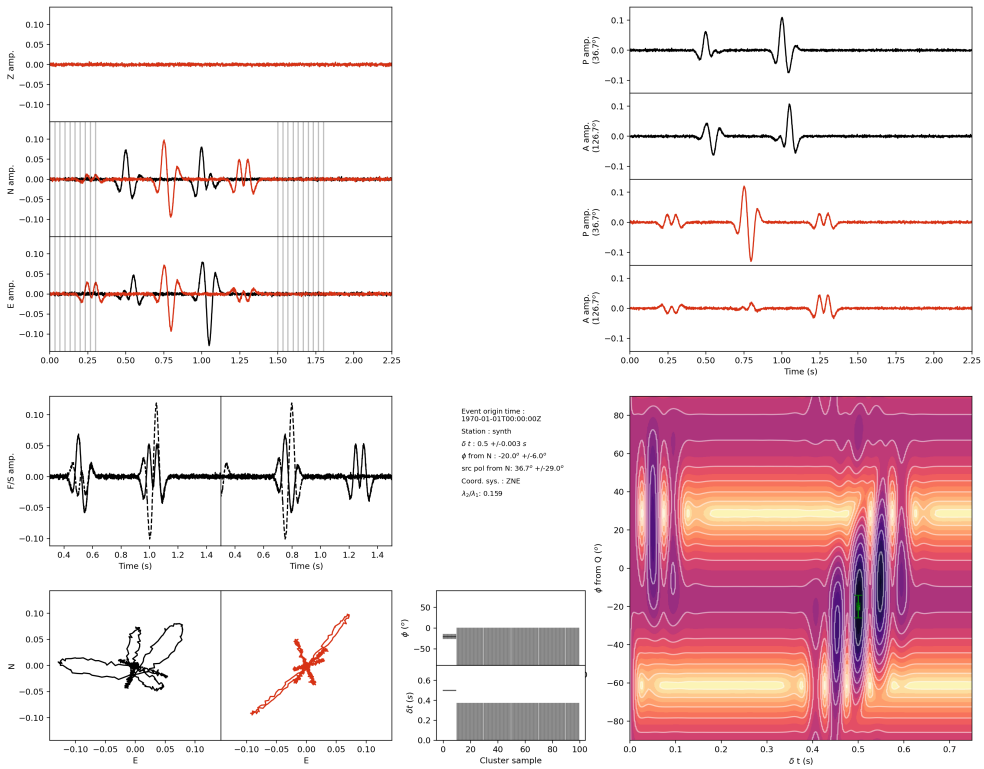
## And measure effective splitting, assuming single layer, for comparison:

In [19]:

```
# Measure splitting:
event_uid = "multi-layer-effective"
S_phase_arrival_times = [ZNE_st_layer1and2[0].stats.starttime+t_src]
back_azis_all_stations = [back_azi]
receiver_inc_angles_all_stations = [event_inclin_angle_at_station]
splitting_event = swspy.splitting.create_splitting_object(ZNE_st_layer1and2, 
splitting_event.overall_win_start_pre_fast_S_pick = 0.5
splitting_event.win_S_pick_tolerance = 0.2
splitting_event.overall_win_start_post_fast_S_pick = 1.0
splitting_event.rotate_step_deg = 2 #1.0
splitting_event.max_t_shift_s = 0.75 #1.0
splitting_event.n_win = 10
splitting_event.perform_sws_analysis(coord_system="ZNE", sws_method="EV")

# And plot splitting result:
splitting_event.plot(outdir=os.path.join("outputs", "plots"))

# And save result to file:
splitting_event.save_result(outdir=os.path.join("outputs", "data"))
```




```
/Users/eart0504/Documents/python/github_repositories/swspy/swspy/splitting/spl
it.py:1655: UserWarning: constrained_layout not applied.  At least one axes co
llapsed to zero width or height.
  plt.savefig(os.path.join(outdir, ''.join((self.event_uid, "_", station, ".pn
g"))), dpi=300)
Saved sws result to: outputs/data/multi-layer-effective_sws_result.csv
```

In [ ]:

In [ ]:

In [ ]: