

```
In [ ]: # =====
# GOOGLE COLAB SETUP / GOOGLE COLAB SETUP
# =====

# Sjekk om vi kjører i Google Colab
try:
    import google.colab
    IN_COLAB = True
    print("🔧 Kjører i Google Colab – installerer avhengigheter...")
    print("🔧 Running in Google Colab – installing dependencies...")

    # Installer nødvendige pakker
    %pip install -q networkx matplotlib plotly pydantic pyyaml pandas ipywidgets

    # Klon repository
    %git clone https://github.com/arvidl/slektstre.git
    import sys
    sys.path.append('/content/slektstre/src')

except ImportError:
    IN_COLAB = False
    print("🏠 Kjører lokalt / Running locally")
    import sys
    sys.path.append('../src')

print(f"📍 Miljø: {'Google Colab' if IN_COLAB else 'Lokal'}")
print(f"📍 Environment: {'Google Colab' if IN_COLAB else 'Local'}")
```

## Visualisering av slektstre

I denne notebooken utforsker vi alle tilgjengelige visualiseringsalternativer for slektstre.

### Tilgjengelige visualiseringer

1. **Hierarkisk tre** - Tradisjonell struktur
2. **Fan chart** - Sirkulær visning
3. **Interaktiv tre** - Plotly-basert
4. **Hourglass view** - Fokuspersone i midten
5. **Statistikk** - Grafer og diagrammer

```
In [1]: # Importer nødvendige biblioteker
import sys
sys.path.append('../src')

from models import Person, Ekteskap, FamilieData, Gender
from tree import Slektstre
from family_io import load_from_yaml
from visualization import (
    plot_hierarchical_tree,
```

```

    plot_fan_chart,
    plot_interactive_tree,
    plot_statistics,
    plot_hourglass_view
)
import matplotlib.pyplot as plt
import plotly.express as px
from datetime import date

print("✅ Alle biblioteker importert!")

```

✅ Alle biblioteker importert!

```

In [2]: # Last eksempel-familie
familie_data = load_from_yaml('../data/eksempel_familie.yaml')
slektstre = Slektstre(familie_data)

print(f"Familie lastet med {len(familie_data.personer)} personer")
print(f"Beskrivelse: {familie_data.beskrivelse}")

# Vis statistikk
stats = slektstre.get_statistics()
print(f"\n📊 Statistikk:")
print(f"Antall generasjoner: {stats['max_generation'] + 1}")
print(f"Gjennomsnittsalder: {stats['average_age']:.1f} år")
print(f"Antall ekteskap: {stats['total_marriages']}")

```

Familie lastet med 17 personer

Beskrivelse: Eksempel familie med 4 generasjoner – Lundervold familien

📊 Statistikk:  
 Antall generasjoner: 3  
 Gjennomsnittsalder: 49.2 år  
 Antall ekteskap: 5

## 1. Hierarkisk tre

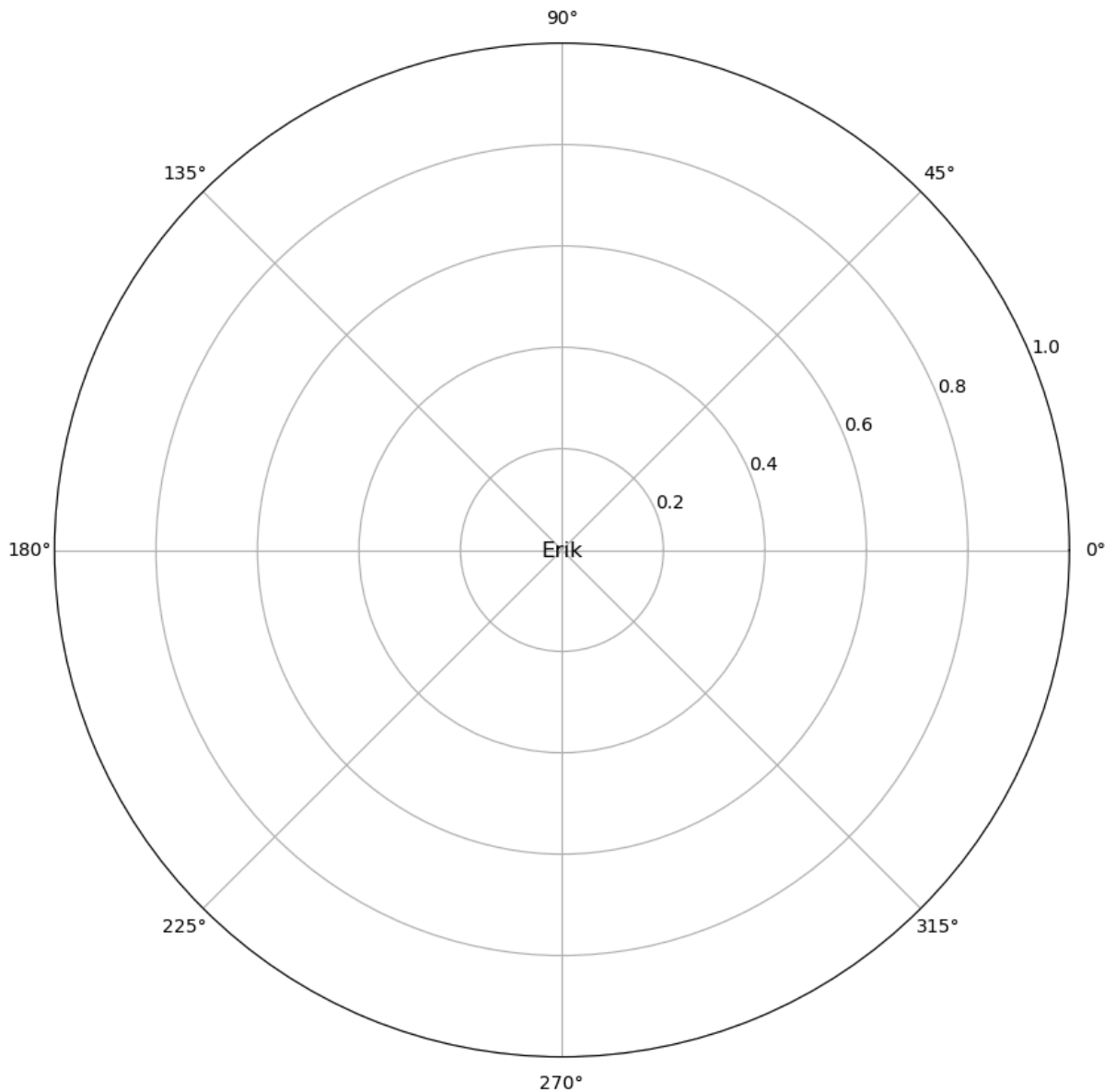
Tradisjonell struktur med generasjoner fra topp til bunn:

```

In [3]: # Plott hierarkisk tre
fig = plot_hierarchical_tree(slektstre, title="Hierarkisk slektstre")
plt.show()

```





### 3. Interaktiv tre

Plotly-basert interaktiv visualisering med hover-info:

```
In [5]: # Plott interaktiv tre
fig = plot_interactive_tree(slektstre, title="Interaktiv slektstre")
fig.show()
```

### 4. Hourglass View

Fokuspersone i midten med forfedre over og etterkommere under:

```
In [6]: # Velg en fokuspersone (Arvid Lundervold)
fokuspersone_id = "p3" # Arvid Lundervold
fokuspersone = slektstre.get_persone(fokuspersone_id)
```

```

if fokusperson:
    print(f"Fokusperson: {fokusperson.fullt_navn} (ID: {fokusperson.id})")

    # Plott hourglass view
    fig = plot_hourglass_view(slektstre, fokusperson_id, title=f"Hourglass View")
    plt.show()
else:
    print("Fokusperson ikke funnet!")

```

Fokusperson: Arvid Lundervold (ID: p3)

### Hourglass View - Arvid Lundervold



## 5. Statistikk og diagrammer

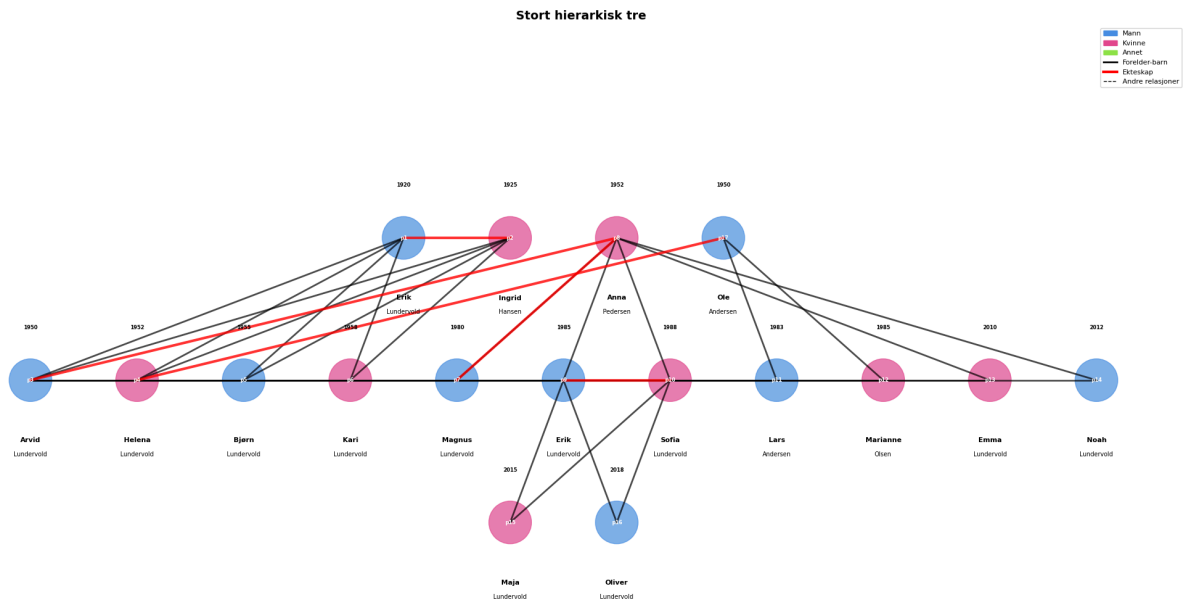
Vis ulike statistikk-diagrammer:

```
In [7]: # Plott statistikk
fig = plot_statistics(slektstre)
fig.show()
```

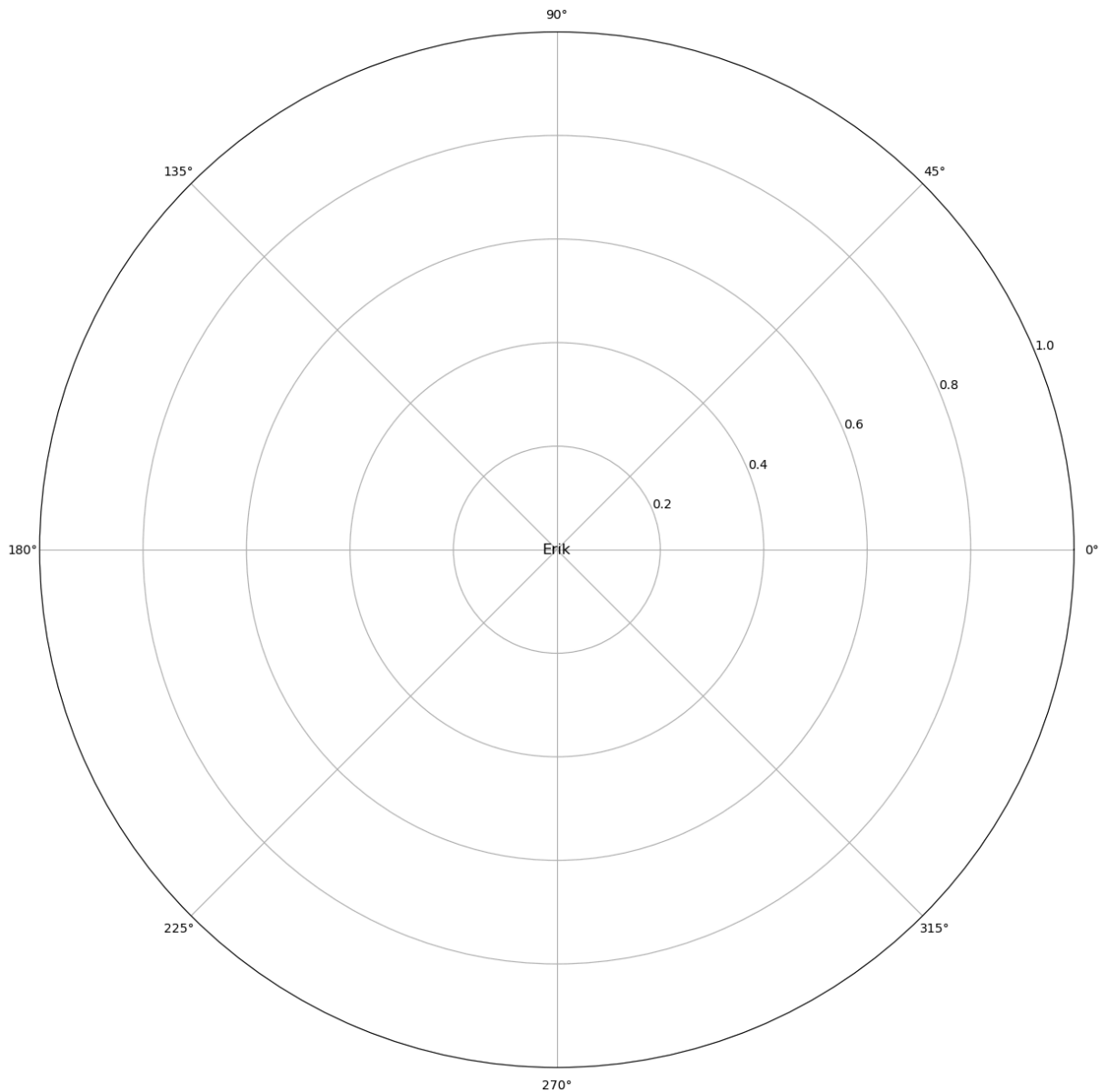
## 6. Tilpassede visualiseringer

La oss eksperimentere med forskjellige parametere:

```
In [8]: # Hierarkisk tre med større figurstørrelse
fig = plot_hierarchical_tree(
    slektstre,
    title="Stort hierarkisk tre",
    figsize=(20, 12)
)
plt.show()
```



```
In [9]: # Fan chart med større figsize
alle_personer = slektstre.get_all_persons()
if alle_personer:
    root_person_id = alle_personer[0].id
    fig = plot_fan_chart(
        slektstre,
        root_person_id,
        title="Stor fan chart",
        figsize=(15, 15)
    )
    plt.show()
else:
    print("Ingen personer funnet i slektstreet!")
```



## 7. Sammenligning av visualiseringer

La oss sammenligne forskjellige visualiseringer side ved side:

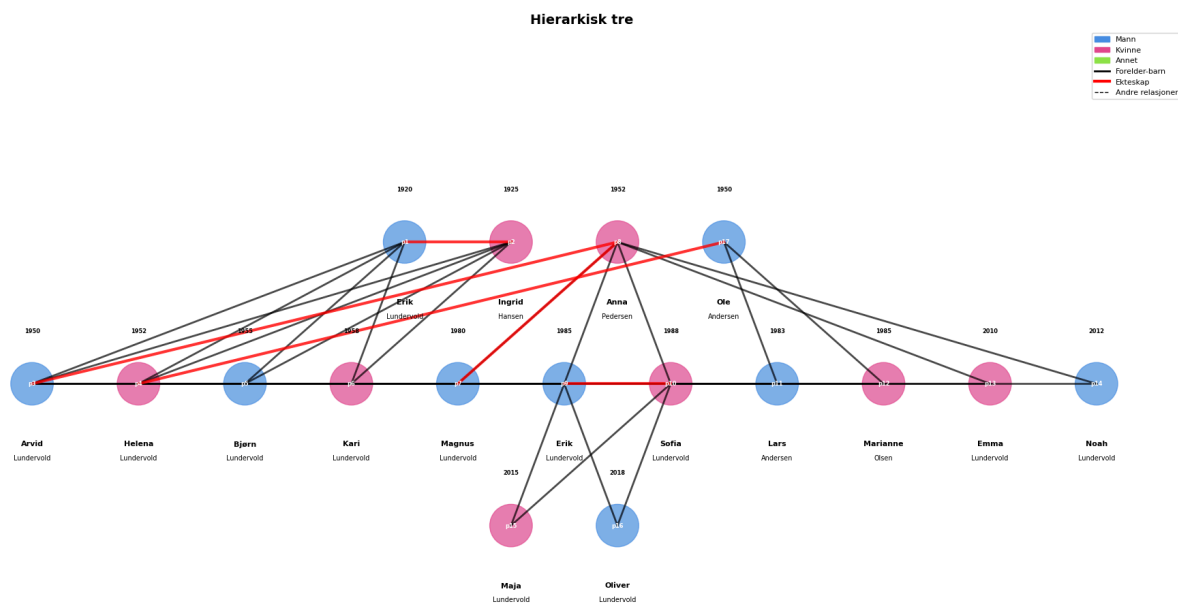
```
In [10]: # Sammenlign alle visualiseringer - hver i sin egen figur
import matplotlib.pyplot as plt

# Hierarkisk tre
fig1 = plot_hierarchical_tree(slektstre, title="Hierarkisk tre")
plt.show()

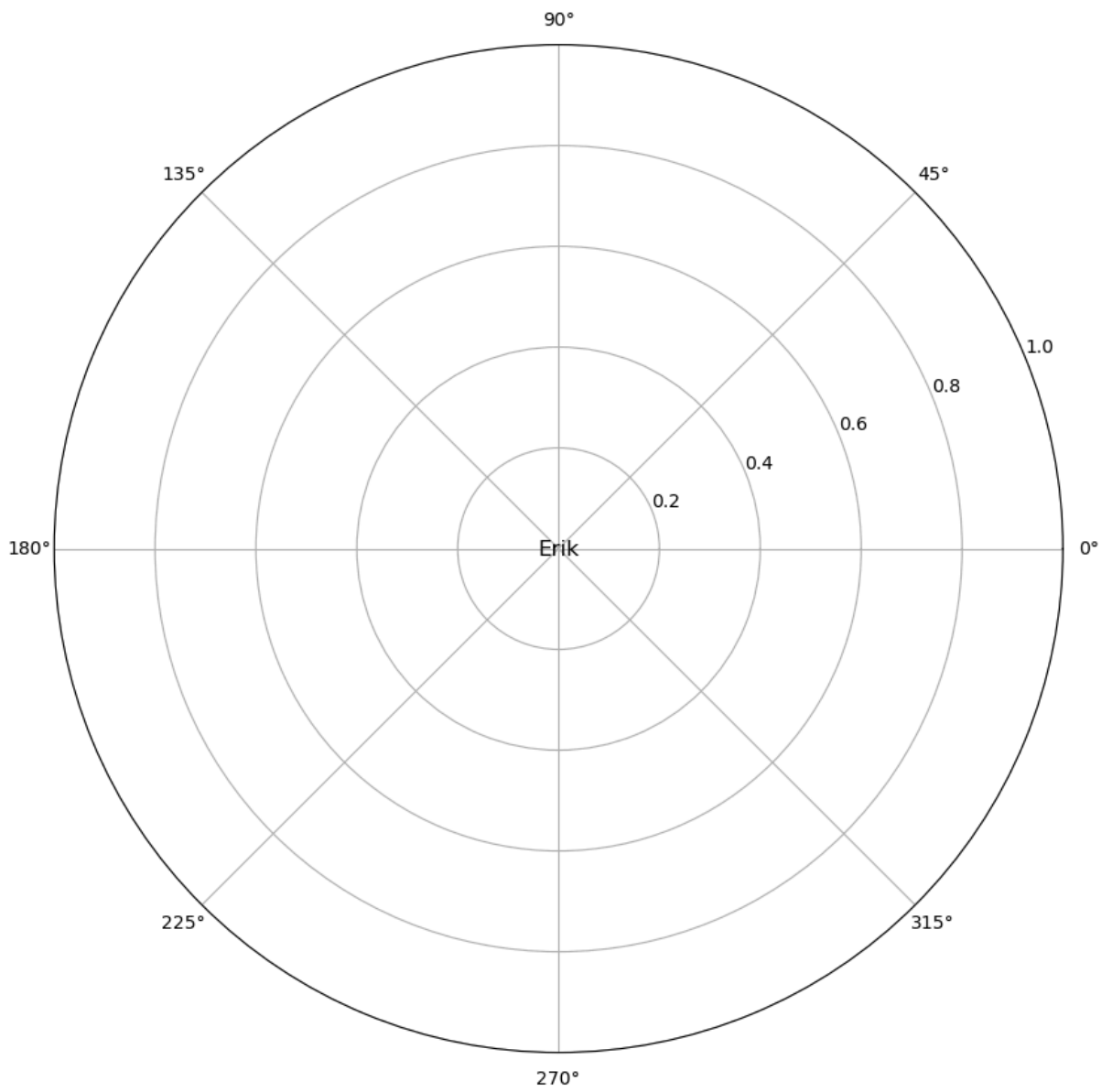
# Fan chart
alle_personer = slektstre.get_all_persons()
if alle_personer:
    root_person_id = alle_personer[0].id
    fig2 = plot_fan_chart(slektstre, root_person_id, title="Fan Chart")
    plt.show()
```

```
# Hourglass view
fig3 = plot_hourglass_view(slektstre, "p3", title="Hourglass View")
plt.show()

# Statistikk
fig4 = plot_statistics(slektstre)
fig4.show()
```







## Hourglass View



## 8. Eksport av visualiseringer

La oss lagre visualiseringer til filer:

```
In [11]: # Lagre visualiseringer til filer
import os

# Opprett mappe for eksporterte bilder
os.makedirs("eksporterte_bilder", exist_ok=True)

# Lagre hierarkisk tre
fig = plot_hierarchical_tree(slektstre, title="Hierarkisk slektstre")
fig.savefig("eksporterte_bilder/hierarkisk_tre.png", dpi=300, bbox_inches='tight')
plt.close(fig)

# Lagre fan chart
```

```

alle_personer = slektstre.get_all_persons()
if alle_personer:
    root_person_id = alle_personer[0].id
    fig = plot_fan_chart(slektstre, root_person_id, title="Fan Chart")
    fig.savefig("eksporterte_bilder/fan_chart.png", dpi=300, bbox_inches='tight')
    plt.close(fig)

# Lagre hourglass view
fig = plot_hourglass_view(slektstre, "p3", title="Hourglass View")
fig.savefig("eksporterte_bilder/hourglass_view.png", dpi=300, bbox_inches='tight')
plt.close(fig)

# Lagre statistikk (Plotly-figur)
fig = plot_statistics(slektstre)
fig.write_image("eksporterte_bilder/statistikk.png", width=800, height=600,
# Plotly-figurer trenger ikke plt.close()

print("✅ Alle visualiseringer lagret til 'eksporterte_bilder/' mappen")
print("📁 Filene:")
for fil in os.listdir("eksporterte_bilder"):
    print(f"    - {fil}")

```

```

✅ Alle visualiseringer lagret til 'eksporterte_bilder/' mappen
📁 Filene:
- fan_chart.png
- hourglass_view.png
- hierarkisk_tre.png
- statistikk.png

```

## 9. Rydde opp

La oss slette de eksporterte bildene:

```

In [12]: # Slett eksporterte bilder
import shutil

if os.path.exists("eksporterte_bilder"):
    shutil.rmtree("eksporterte_bilder")
    print("🗑️ Slettet 'eksporterte_bilder/' mappen")

print("✅ Opprydding fullført!")

```

```






🗑️ Slettet 'eksporterte_bilder/' mappen
✅ Opprydding fullført!

```

## Oppsummering

I denne notebooken har du utforsket alle visualiseringsalternativer:

- ✅ **Hierarkisk tre** - Tradisjonell struktur, god for oversikt
- ✅ **Fan chart** - Sirkulær visning, visuelt tiltalende
- ✅ **Interaktiv tre** - Plotly-basert, zoom og hover-info
- ✅ **Hourglass view** - Fokusperson i midten, god for detaljer

5.  **Statistikk** - Grafer og diagrammer, dataanalyse
6.  **Tilpassede visualiseringer** - Forskjellige størrelser og parametere
7.  **Sammenligning** - Side ved side visning
8.  **Eksport** - Lagre til PNG-filer
9.  **Opprydding** - Slette midlertidige filer

**Anbefalinger:**

- Bruk **hierarkisk tre** for generell oversikt
- Bruk **fan chart** for presentasjoner
- Bruk **interaktiv tre** for utforskning
- Bruk **hourglass view** for fokus på en person
- Bruk **statistikk** for dataanalyse

**Neste steg:** Du har nå fullført alle notebookene! Du kan begynne å bygge ditt eget slektstre.