In [1]:
```
# --- SETUP: load word vectors and define helpers ---
import gensim.downloader as api
from scipy.spatial import distance
import numpy as np
from textwrap import fill

# Load a small, fast model for demos (50d)
model = api.load("glove-wiki-gigaword-50")


def cos_sim(a, b):
    return 1 - distance.cosine(a, b)


def cos_dist(a, b):
    return distance.cosine(a, b)


def explain(text, width=92):
    print(fill(text, width=width))
    print()


explain(
    "How to read cosine measures: cosine similarity ranges from −1 to 1, where 1.0 means two vector
    "point in the same direction (very strong association), 0.0 means no directional association, a
    "−1.0 means opposite directions. Some tools report cosine distance = 1 − cosine similarity; "
    "so a small distance (e.g., 0.14) implies a large similarity (0.86). As a rough guide, similari
    "≥ 0.70 are often read as high, 0.40–0.69 as moderate, and ≤ 0.39 as low, though thresholds var
)
```

How to read cosine measures: cosine similarity ranges from −1 to 1, where 1.0 means two vectors point in the same direction (very strong association), 0.0 means no directional association, and −1.0 means opposite directions. Some tools report cosine distance = 1 − cosine similarity; so a small distance (e.g., 0.14) implies a large similarity (0.86). As a rough guide, similarities ≥ 0.70 are often read as high, 0.40–0.69 as moderate, and ≤ 0.39 as low, though thresholds vary by model.

In [2]:
```
# --- Association network between related concepts ---
probe = model["king"] - model["man"] + model["woman"]
sim = cos_sim(probe, model["queen"])
dist = cos_dist(probe, model["queen"])

print(
    f"Cosine distance between 'king −man + woman' and 'queen') = {dist:.3f}  ->  similarity = {sim:
)
explain(
    "Explanation: Take the direction from man→king and add it to woman. "
    "If the geometry encodes relational regularities, the result should land near 'queen'. "
    f"A cosine distance of {dist:.3f} (cosine similarity {sim:.3f}) indicates a strong association.
)
```

Cosine distance between 'king −man + woman' and 'queen') = 0.139 -> similarity = 0.861

Explanation: Take the direction from man→king and add it to woman. If the geometry encodes relational regularities, the result should land near 'queen'. A cosine distance of 0.139 (cosine similarity 0.861) indicates a strong association.

In [3]:
```
# --- GENDER AXIS + OCCUPATION PROJECTIONS ---
gender_axis = model["man"] - model["woman"]
gender_axis = gender_axis / np.linalg.norm(gender_axis)

occupations = [
    "engineer",
    "scientist",
    "lawyer",
    "programmer",
    "nurse",
    "teacher",
    "director",
    "receptionist",
    "officer",
    "policymaker",
    "cook",
    "hairstylist",
    "veterinarian",
]

rows = []
for w in occupations:
    if w in model:
        v = model[w]
        proj = np.dot(v / np.linalg.norm(v), gender_axis)
        rows.append((w, proj))

rows_sorted = sorted(rows, key=lambda x: x[1], reverse=True)

print("Projection on the gender axis (man − woman):")
for w, p in rows_sorted:
    print(f"  {w:>12s}  {p:+.3f}")
print()

explain(
    "A single 'gender direction' is defined as the vector from 'woman' to 'man'. "
    "Projecting words on this axis quantifies corpus-coded gender alignment: positive values "
    "lean male-coded, negative values female-coded."
)
```

Projection on the gender axis (man − woman):
```
      director  +0.177
       officer  +0.138
  policymaker  +0.108
      engineer  +0.080
    programmer  +0.066
     scientist  +0.052
          cook  +0.028
        lawyer  -0.020
 veterinarian  -0.166
       teacher  -0.179
   hairstylist  -0.204
  receptionist  -0.331
         nurse  -0.380
```

A single 'gender direction' is defined as the vector from 'woman' to 'man'. Projecting words on this axis quantifies corpus-coded gender alignment: positive values lean male-coded, negative values female-coded.

In [4]:
```
# --- MINI WEAT (SCIENCE/ARTS × MALE/FEMALE) ---
science = [
    "science",
    "technology",
    "physics",
    "chemistry",
    "einstein",
    "nasa",
    "experiment",
    "astronomy",
]
arts = [
    "poetry",
    "art",
    "dance",
    "literature",
    "novel",
    "symphony",
    "drama",
    "sculpture",
]
male = ["man", "male", "boy", "brother", "he", "him", "his", "son"]
female = ["woman", "female", "girl", "sister", "she", "her", "hers", "daughter"]


def set_assoc(X, A, B):
    # average similarity to set A minus average similarity to set B
    simsA = [
        cos_sim(model[x], model[a]) for x in X for a in A if x in model and a in model
    ]
    simsB = [
        cos_sim(model[x], model[b]) for x in X for b in B if x in model and b in model
    ]
    return np.mean(simsA) - np.mean(simsB)


def weat_effect(X, Y, A, B):
    sX = [set_assoc([x], A, B) for x in X if x in model]
    sY = [set_assoc([y], A, B) for y in Y if y in model]
    # pooled std
    pooled = np.std(sX + sY, ddof=1)
    return (np.mean(sX) - np.mean(sY)) / pooled if pooled > 0 else float("nan")


effect = weat_effect(science, arts, male, female)
print(f"Mini-WEAT effect size (science-male vs arts-female): {effect:.2f}\n")

explain(
    "WEAT (Word Embedding Association Test): A positive WEAT effect size means 'science' terms alig
    "and 'arts' terms align more with female than with male. This quantifies culturally patterned "
    "associations encoded in the feature space."
)
```

Mini-WEAT effect size (science-male vs arts-female): 1.61

WEAT (Word Embedding Association Test): A positive WEAT effect size means 'science' terms align more with male words than with female, and 'arts' terms align more with female than with male. This quantifies culturally patterned associations encoded in the feature space.

In [5]:
```
# ---- POLYSEMY DEMO: 'bank' as finance vs river ----
targets = ["bank"]

tilt_to_finance = (model["money"] + model["loan"] + model["finance"]) / 3
tilt_to_river   = (model["river"] + model["stream"] + model["water"]) / 3

def nearest(word_vec, k=8, banned=("bank", "banks")):
    sims = []
    banned = {b.lower() for b in banned}
    for w in list(model.index_to_key)[:50000]:  # cap for speed
        if w.lower() in banned:
            continue
        sims.append((w, cos_sim(word_vec, model[w])))
    sims.sort(key=lambda x: x[1], reverse=True)
    return [w for w, s in sims[:k]]

base = model["bank"]
fin  = base + 0.6 * (tilt_to_finance - base)
geo  = base + 0.6 * (tilt_to_river - base)

print("Nearest neighbours of 'bank' (base):", nearest(base))
print("Nearest neighbours of 'bank' (tilted toward finance):", nearest(fin))
print("Nearest neighbours of 'bank' (tilted toward river):", nearest(geo))
print()

explain(
    "POLYSEMY DEMO: Small directional tilts move 'bank' between financial and geographical neighbou
    "This shows that meaning is not a single essence but a recombination of partial traces,\n"
    "assembled by context — an example of dividual sense composition."
)
```

```
Nearest neighbours of 'bank' (base): ['securities', 'banking', 'investment', 'exchange', 'financia
l', 'credit', 'lender', 'capital']
Nearest neighbours of 'bank' (tilted toward finance): ['credit', 'funds', 'fund', 'investment', 'loa
ns', 'loan', 'financial', 'finance']
Nearest neighbours of 'bank' (tilted toward river): ['flows', 'shore', 'river', 'along', 'from', 'ce
ntral', 'stream', 'through']

POLYSEMY DEMO: Small directional tilts move 'bank' between financial and geographical
neighbourhoods. This shows that meaning is not a single essence but a recombination of
partial traces, assembled by context — an example of divisual sense composition.
```

In [ ]: