In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

import scipy.stats as stats

def t_test(df1, df2, target="SOG"):
    t_stat, p_value = stats.ttest_ind(df1[target].dropna(), df2[target].dropna())
    print(f"T-statistic: {t_stat:.3f}, p-value: {p_value:.15f}")

    # If p-value is less than 0.05, the difference is statistically significant
    if p_value < 0.05:
        print("The difference is statistically significant, keeping data split.")
    else:
        print("The difference is not statistically significant, keeping data combined.")
```

In [2]:
```
df = pd.read_csv("all_data.csv")
```

In [3]:
```
senseboard_9juin = df[
    (df["boat_name"] == "SenseBoard") &
    (df["opponent_name"] == "Karl Maeder") &
    (df["ISODateTimeUTC"].str.startswith("2025-06-09"))]
```

Upwind Data of Gian on the senseboard on the 9th of June

In [4]:
```
# Upwind data
senseboard_9juin_upwind = senseboard_9juin[senseboard_9juin["TWA"]>0]

nb_poids_differents_upwind = senseboard_9juin_upwind["boat_weight"].nunique()

print(f"Nombre de poids différents utilisés par Senseboard (Gian) le 9 juin : {nb_poids_differents_upwind}")
poids_uniques_upwind = senseboard_9juin_upwind["boat_weight"].unique()
print(f"Poids uniques: {poids_uniques_upwind}\n")

if len(poids_uniques_upwind) != 2:
    print("Erreur : il faut exactement deux poids pour effectuer le t-test.")
else:
    poids1_upwind, poids2_upwind = poids_uniques_upwind

    groupe1_df_upwind = senseboard_9juin_upwind[senseboard_9juin_upwind["boat_weight"] == poids1_upwind]
    groupe2_df_upwind = senseboard_9juin_upwind[senseboard_9juin_upwind["boat_weight"] == poids2_upwind]
    t_test(groupe1_df_upwind, groupe2_df_upwind)

print(f"\nNumber of individuals in Group 1: {len(groupe1_df_upwind)}")
print(f"Number of individuals in Group 2: {len(groupe2_df_upwind)}")

print(f"\nWeight in Group 1: {poids1_upwind}, Average SOG: {groupe1_df_upwind['SOG'].mean()}")
print(f"Weight in Group 2: {poids2_upwind}, Average SOG: {groupe2_df_upwind['SOG'].mean()}")
```

```
Nombre de poids différents utilisés par Senseboard (Gian) le 9 juin : 2
Poids uniques: [109.09 115.09]

T-statistic: 5.647, p-value: 0.000000017156318
The difference is statistically significant, keeping data split.

Number of individuals in Group 1: 2683
Number of individuals in Group 2: 2588

Weight in Group 1: 109.09, Average SOG: 23.3155795751025
Weight in Group 2: 115.09, Average SOG: 23.180486862442038
```

Downwind Data of Gian on the senseboard on the 9th of June

In [5]:

```
# Downwind data
senseboard_9juin_downwind = senseboard_9juin[senseboard_9juin["TWA"]<=0]

nb_poids_differents_downwind = senseboard_9juin_downwind["boat_weight"].nunique()

print(f"Nombre de poids différents utilisés par Senseboard (Gian) le 9 juin : {nb_poids_differents_downwind}")
poids_uniques_downwind = senseboard_9juin_downwind["boat_weight"].unique()
print(f"Poids uniques: {poids_uniques_downwind}\n")

if len(poids_uniques_downwind) != 2:
    print("Erreur : il faut exactement deux poids pour effectuer le t-test.")
else:
    poids1_downwind, poids2_downwind = poids_uniques_downwind

    groupe1_df_downwind = senseboard_9juin_downwind[senseboard_9juin_downwind["boat_weight"] == poids1_downwind]
    groupe2_df_downwind = senseboard_9juin_downwind[senseboard_9juin_downwind["boat_weight"] == poids2_downwind]
    t_test(groupe1_df_downwind, groupe2_df_downwind)

print(f"\nNumber of individuals in Group 1: {len(groupe1_df_downwind)}")
print(f"Number of individuals in Group 2: {len(groupe2_df_downwind)}")

print(f"\nWeight in Group 1: {poids1_downwind}, Average SOG: {groupe1_df_downwind['SOG'].mean()}")
print(f"Weight in Group 2: {poids2_downwind}, Average SOG: {groupe2_df_downwind['SOG'].mean()}")
```

```
Nombre de poids différents utilisés par Senseboard (Gian) le 9 juin : 2
Poids uniques: [109.09 115.09]

T-statistic: 18.690, p-value: 0.000000000000000
The difference is statistically significant, keeping data split.

Number of individuals in Group 1: 1257
Number of individuals in Group 2: 2034

Weight in Group 1: 109.09, Average SOG: 27.65552903739061
Weight in Group 2: 115.09, Average SOG: 26.918731563421826
```

General Data of Gian on the senseboard on the 9th of June; simple biaised mean and balanced means

In [6]:

```
# --- 1. Identify the two weight configurations used ---
unique_weights = sorted(senseboard_9juin["boat_weight"].unique())
num_weights = len(unique_weights)
```

```
print(f"Number of distinct weights used on June 9th: {num_weights}")
print(f"Weight values: {unique_weights}\n")

if num_weights != 2:
    print("Error: This analysis requires exactly two weight configurations.")
else:
    light_weight, heavy_weight = unique_weights

    # --- 2. Global analysis without directional separation ---
    group_light = senseboard_9juin[senseboard_9juin["boat_weight"] == light_weight]
    group_heavy = senseboard_9juin[senseboard_9juin["boat_weight"] == heavy_weight]

    print("\n--- T-test on overall (unstratified) data ---")
    t_test(group_light, group_heavy)

    mean_light = group_light["SOG"].mean()
    mean_heavy = group_heavy["SOG"].mean()
    n_light = len(group_light)
    n_heavy = len(group_heavy)
    total = n_light + n_heavy
    pct_light = 100 * n_light / total
    pct_heavy = 100 * n_heavy / total

    print("\n--- Global statistics ---")
    print(f"Light group: {n_light} samples ({pct_light:.1f}%), weight = {light_weight}, average SOG = {mean_light:.3f}")
    print(f"Heavy group: {n_heavy} samples ({pct_heavy:.1f}%), weight = {heavy_weight}, average SOG = {mean_heavy:.3f}")

    # --- 3. Directional breakdown with weighted average (50% upwind / 50% downwind) ---
    upwind_light = senseboard_9juin_upwind[senseboard_9juin_upwind["boat_weight"] == light_weight]
    upwind_heavy = senseboard_9juin_upwind[senseboard_9juin_upwind["boat_weight"] == heavy_weight]
    downwind_light = senseboard_9juin_downwind[senseboard_9juin_downwind["boat_weight"] == light_weight]
    downwind_heavy = senseboard_9juin_downwind[senseboard_9juin_downwind["boat_weight"] == heavy_weight]

    mean_up_light = upwind_light["SOG"].mean()
    mean_up_heavy = upwind_heavy["SOG"].mean()
    mean_down_light = downwind_light["SOG"].mean()
    mean_down_heavy = downwind_heavy["SOG"].mean()

    sog_light_weighted = 0.5 * (mean_up_light + mean_down_light)
    sog_heavy_weighted = 0.5 * (mean_up_heavy + mean_down_heavy)

    print("\n--- Weighted average by direction (50% upwind / 50% downwind) ---")
    print(f"Light weight SOG (weighted): {sog_light_weighted:.3f}")
    print(f"Heavy weight SOG (weighted): {sog_heavy_weighted:.3f}")

    # --- 4. Balanced sampling: same sample count in each group and direction ---
    n_up = min(len(upwind_light), len(upwind_heavy))
    n_down = min(len(downwind_light), len(downwind_heavy))

    up_light_sample = upwind_light.sample(n=n_up, random_state=42)
    up_heavy_sample = upwind_heavy.sample(n=n_up, random_state=42)
    down_light_sample = downwind_light.sample(n=n_down, random_state=42)
    down_heavy_sample = downwind_heavy.sample(n=n_down, random_state=42)

    sample_light = pd.concat([up_light_sample, down_light_sample])
    sample_heavy = pd.concat([up_heavy_sample, down_heavy_sample])

    print("\n--- Balanced directional sampling (equal number of samples per direction and weight) ---")
```

```python
print(f"Light weight SOG (balanced): {sample_light['SOG'].mean():.3f}")
print(f"Heavy weight SOG (balanced): {sample_heavy['SOG'].mean():.3f}")

print("\n--- T-test on balanced directional samples ---")
t_test(sample_light, sample_heavy)

# --- 5. Detection of Simpson's paradox due to direction imbalance ---
print("\n--- Directional sample counts by weight ---")
print(f"Upwind   - Light: {len(upwind_light)} samples")
print(f"Upwind   - Heavy: {len(upwind_heavy)} samples")
print(f"Downwind - Light: {len(downwind_light)} samples")
print(f"Downwind - Heavy: {len(downwind_heavy)} samples")

total_light = len(upwind_light) + len(downwind_light)
total_heavy = len(upwind_heavy) + len(downwind_heavy)

pct_up_light = 100 * len(upwind_light) / total_light if total_light > 0 else 0
pct_down_light = 100 * len(downwind_light) / total_light if total_light > 0 else 0
pct_up_heavy = 100 * len(upwind_heavy) / total_heavy if total_heavy > 0 else 0
pct_down_heavy = 100 * len(downwind_heavy) / total_heavy if total_heavy > 0 else 0

print("\n--- Proportion of upwind/downwind in each weight group ---")
print(f"Light weight: {pct_up_light:.1f}% upwind, {pct_down_light:.1f}% downwind")
print(f"Heavy weight: {pct_up_heavy:.1f}% upwind, {pct_down_heavy:.1f}% downwind")

heavier_slower_in_up = mean_up_heavy < mean_up_light
heavier_slower_in_down = mean_down_heavy < mean_down_light
heavier_faster_globally = mean_heavy > mean_light

if heavier_slower_in_up and heavier_slower_in_down and heavier_faster_globally:
    print("\nSimpson's paradox detected:")
    print("- In both upwind and downwind segments, the heavier configuration performs worse.")
    print("- Yet, the overall average suggests the heavier configuration is faster.")
    print("- This discrepancy arises from an imbalance in the directional distribution between weight groups.")
    print("- Heavier samples may dominate faster downwind segments, skewing the global average.")
    print("- Directionally stratified or balanced analyses are necessary to remove this bias.")
else:
    print("\nNo Simpson's paradox detected. Direction-specific results align with the overall trend.")
```

```
Number of distinct weights used on June 9th: 2
Weight values: [109.09, 115.09]


--- T-test on overall (unstratified) data ---
T-statistic: -2.676, p-value: 0.007467354374147
The difference is statistically significant, keeping data split.

--- Global statistics ---
Light group: 3940 samples (46.0%), weight = 109.09, average SOG = 24.700
Heavy group: 4622 samples (54.0%), weight = 115.09, average SOG = 24.826

--- Weighted average by direction (50% upwind / 50% downwind) ---
Light weight SOG (weighted): 25.486
Heavy weight SOG (weighted): 25.050

--- Balanced directional sampling (equal number of samples per direction and weight) ---
Light weight SOG (balanced): 24.737
Heavy weight SOG (balanced): 24.415

--- T-test on balanced directional samples ---
T-statistic: 6.639, p-value: 0.000000000033665
The difference is statistically significant, keeping data split.

--- Directional sample counts by weight ---
Upwind   - Light: 2683 samples
Upwind   - Heavy: 2588 samples
Downwind - Light: 1257 samples
Downwind - Heavy: 2034 samples

--- Proportion of upwind/downwind in each weight group ---
Light weight: 68.1% upwind, 31.9% downwind
Heavy weight: 56.0% upwind, 44.0% downwind

Simpson's paradox detected:
- In both upwind and downwind segments, the heavier configuration performs worse.
- Yet, the overall average suggests the heavier configuration is faster.
- This discrepancy arises from an imbalance in the directional distribution between weight groups.
- Heavier samples may dominate faster downwind segments, skewing the global average.
- Directionally stratified or balanced analyses are necessary to remove this bias.
```

In [ ]: