🌬️ Équilibre d'un mât haubané à deux niveaux

⚙️ 1. Données du problème

Le mât est une poutre triangulée en aluminium **tube Ø50×2 mm**, de hauteur **h = 16 m**, haubané à **deux niveaux** :

Paramètre	Symbole
masse de la lampe	m_{lampe}
masse du mât	m_{mat}
Hauteur du mât	ha_{mat}
Diamètre tube	D
Coefficient de traînée du tube	C_d
Distance d'ancrage	I_{anc}
Surface de la lampe	A_{lampe}
Pression du vent	p_{vent}

💨 Répartition du vent

Le vent agit sur :

- **la lampe** (surface frontale),
- **le mât triangulé** (surface projetée équivalente).

Force totale due au vent :

$$F_{vent} = F_{lampe} + F_{mât}$$

avec :

$$F_{lampe} = p_{vent} \times A_{lampe}$$

et

$$F_{mât} = p_{vent} \times D \times h \times C_d$$

💡 Pour un mât treillis, on peut appliquer un **facteur de porosité** (≈0.4) pour ajuster la charge effective.

🧩 2. Angles et géométrie

Les haubans sont disposés à 120° dans le plan horizontal, à deux niveaux :

$$\tan(\alpha_1) = \frac{h}{I_{anc}}, \quad \tan(\alpha_2) = \frac{h/2}{I_{anc}}$$

soit :

$$\alpha_1 = 63.4^\circ, \quad \alpha_2 = 45^\circ$$

🪜 3. Répartition de la charge du vent

Le profil du vent augmente avec la hauteur.
En supposant une loi de puissance $p(z) \propto z^\alpha$ avec $\alpha = 0.2$:

$$\frac{F_{haut}}{F_{total}} = 1 - \left(\frac{1}{2}\right)^{1+\alpha} \approx 0.64$$

On adopte donc une répartition **64 % – 36 %** entre les deux niveaux :

$$F_{vent,haut} = 0.64\,F_{vent}, \quad F_{vent,bas} = 0.36\,F_{vent}$$

🧮 4. Équilibre des forces et des moments

a) Équilibre horizontal (axe du vent) :

$$R_x + \cos(\alpha_1)(T_{1,1} + T_{a,1}) + \cos(\alpha_2)(T_{1,2} + T_{a,2}) = F_{vent}$$

b) Équilibre latéral (perpendiculaire au vent) :

Les haubans non sollicités dans le plan du vent génèrent une composante latérale équilibrée par la réaction au sol R_y :

$$R_y = -\frac{\sqrt{3}}{2}\left[\cos(\alpha_1)(T_{a,1}) + \cos(\alpha_2)(T_{a,2})\right]$$

Cette réaction assure la **stabilité tridimensionnelle** du mât.

c) Équilibre des moments autour du pied :

La résultante du vent agit à une hauteur de **0.6 h** :

$$F_{vent} \times 0.6h = \cos(\alpha_1)(T_{1,1} + T_{a,1})\,h + \cos(\alpha_2)(T_{1,2} + T_{a,2})\,\frac{h}{2}$$

d) Équilibre vertical :

$$R_z = F_{ec} + P_{mât} + 3(T_{1,1}\sin(\alpha_1) + T_{1,2}\sin(\alpha_2))$$

🧷 5. Hypothèses de répartition

Dans chaque niveau :

$$T_{1,i} = 2\,T_{a,i}$$

ce qui correspond à un hauban **dans l'axe du vent** deux fois plus chargé que les deux latéraux.

Ainsi :

$$\begin{cases} T_{1,1} = \dfrac{4F_{vent,haut}}{9\cos(\alpha_1)} \\[2ex] T_{a,1} = \dfrac{2F_{vent,haut}}{9\cos(\alpha_1)} \\[2ex] T_{1,2} = \dfrac{2F_{vent,bas}}{9\cos(\alpha_2)} \\[2ex] T_{a,2} = \dfrac{F_{vent,bas}}{9\cos(\alpha_2)} \\[2ex] R_x = -\dfrac{1}{2}\cos(\alpha_2)(T_{1,2} + T_{a,2}) \\[2ex] R_y = -\dfrac{\sqrt{3}}{2}[\cos(\alpha_1)T_{a,1} + \cos(\alpha_2)T_{a,2}] \end{cases}$$

📊 6. Interprétation physique

- Les **haubans supérieurs** reprennent environ **64 %** de la charge du vent.
- Les **haubans inférieurs** stabilisent le mât (≈36 % du vent).
- Les **câbles dans l'axe du vent** sont les plus tendus.
- La **réaction horizontale** R_x peut être nulle si le pied est **articulé**.
- La **réaction latérale** R_y équilibre les efforts hors plan.
- La **réaction verticale** R_z inclut le poids du mât et la lampe.

🧠 7. Points à vérifier ou affiner

⚙️ Répartition du vent (0.64 – 0.36)

Issue d'un profil de vent réaliste $p(z) \propto z^{0.2}$.
La différence avec 70–30 est négligeable pour un dimensionnement préliminaire.

⚙️ Moment d'équilibre

Le moment de chaque câble s'écrit :

$$M_i = T_i \cos(\alpha_i)\, z_i$$

et la résultante du vent agit à **0.6 h** (centre de poussée).

⚙️ Réactions horizontales et latérales

- R_x : réaction dans le plan du vent, transmise si le pied est encastré.
- R_y : réaction latérale, équilibre les efforts transversaux dans le plan perpendiculaire au vent.

⚙️ Effort vertical R_z

Inclut le poids du mât et la précontrainte des haubans :

$$R_z = P_{\text{mât}} + P_{lampe} + 3(T_{1,1}\sin(\alpha_1) + T_{1,2}\sin(\alpha_2))$$

⚙️ Vérification dimensionnelle

Lien avec la pression dynamique du vent :

$$F_{vent} = \tfrac{1}{2}\rho C_d A V^2$$

avec $C_d \approx 1.2$ pour un tube cylindrique, $\rho \approx 1.225\,\mathrm{kg/m^3}$.

📈 8. Visualisation Python

Une cellule Python peut :

- définir les paramètres $h, I_{anc}, C_d, p_{vent}$, etc.
- calculer toutes les tensions et réactions ($T_{1,i}$, $T_{a,i}$, R_x, R_y, R_z),
- tracer le mât et les haubans avec flèches de vent,
- et afficher un **résumé numérique automatique**.

⚖️ 9. Décomposition 3D des efforts dans les haubans

🎯 Objectif

Chaque hauban est incliné d'un angle α_i avec l'horizontale et orienté à **120°** les uns des autres dans le plan horizontal.
On veut exprimer les **composantes cartésiennes** (X, Y, Z) des efforts de tension $T_{1,i}$ (dans le plan du vent) et $T_{a,i}$ (les deux autres).

🧭 9.1 Orientation des haubans

On définit l'axe :

- X : **axe du vent**,
- Y : **perpendiculaire au vent**,
- Z : **vertical (positif vers le haut)**.

Les trois haubans de chaque niveau i sont orientés à :

$$\theta_{1,i} = 0^\circ, \quad \theta_{2,i} = 120^\circ, \quad \theta_{3,i} = 240^\circ$$

9.2 Composantes d'un hauban

Pour un hauban de tension T_i incliné d'un angle α_i :

$$\begin{cases} T_x = T_i \cos(\alpha_i)\cos(\theta_i) \\ T_y = T_i \cos(\alpha_i)\sin(\theta_i) \\ T_z = -T_i \sin(\alpha_i) \end{cases}$$

> 🔸 Le signe « – » en T_z traduit que le hauban tire **vers le bas** le sommet du mât.

9.3 Application par niveau

Niveau supérieur ($i = 1$, α_1)

Hauban	Tension	θ (°)	T_x	T_y	T_z
Hauban avant (vent)	$T_{1,1}$	0	$T_{1,1}\cos\alpha_1$	0	$-T_{1,1}\sin\alpha_1$
Hauban latéral G	$T_{a,1}$	120	$-\frac{1}{2}T_{a,1}\cos\alpha_1$	$\frac{\sqrt{3}}{2}T_{a,1}\cos\alpha_1$	$-T_{a,1}\sin\alpha_1$
Hauban latéral D	$T_{a,1}$	240	$-\frac{1}{2}T_{a,1}\cos\alpha_1$	$-\frac{\sqrt{3}}{2}T_{a,1}\cos\alpha_1$	$-T_{a,1}\sin\alpha_1$

Niveau inférieur ($i = 2$, α_2)

Hauban	Tension	θ (°)	T_x	T_y	T_z
Hauban avant (vent)	$T_{1,2}$	0	$T_{1,2}\cos\alpha_2$	0	$-T_{1,2}\sin\alpha_2$
Hauban latéral	$T_{a,2}$	120	$-\frac{1}{2}T_{a,2}\cos\alpha_2$		$-T_{a,2}\sin\alpha_2$

Hauban	Tension	θ (°)	T_x	T_y	T_z
G				$\frac{\sqrt{3}}{2}T_{a,2}\cos\alpha_2$	
Hauban latéral D	$T_{a,2}$	240	$-\frac{1}{2}T_{a,2}\cos\alpha_2$	$-\frac{\sqrt{3}}{2}T_{a,2}\cos\alpha_2$	$-T_{a,2}\sin\alpha_2$

📊 9.4 Somme vectorielle des efforts par niveau

En sommant les trois haubans d'un même niveau :

$$\boxed{\begin{aligned}\sum T_x &= \cos(\alpha_i)(T_{1,i} - T_{a,i}) \\ \sum T_y &= 0 \\ \sum T_z &= -\sin(\alpha_i)(T_{1,i} + 2T_{a,i})\end{aligned}}$$

Ces résultats confirment :

- **Effort horizontal** dans le plan du vent → proportionnel à $(T_{1,i} - T_{a,i})$
- **Aucune résultante latérale (Y)** (symétrie)
- **Effort vertical** descendant dû à la tension des haubans.

⚖️ 9.5 Équilibre global 3D du mât

En notant la somme sur les deux niveaux :

$$\boxed{\begin{aligned}R_x + \sum_i \sum T_{x,i} = F_{vent} \\ R_y + \sum_i \sum T_{y,i} = 0 \\ R_z + \sum_i \sum T_{z,i} + P_{\text{mât}} + P_{lampe} = 0\end{aligned}}$$

où chaque $\sum T_{x,i}$, $\sum T_{y,i}$, $\sum T_{z,i}$ est défini comme ci-dessus.

🧠 9.6 Interprétation

- Les haubans **dans le plan du vent** apportent la résistance principale (T_x positif vers l'arrière).
- Les haubans **latéraux** assurent la **stabilité transversale** (T_y équilibrés).
- Tous les haubans contribuent à la **compression verticale** du mât (T_z négatif).

In [1]:
```
import sympy as sp
import matplotlib.pyplot as plt
```

```python
import numpy as np
```

In [2]:

```python
# ================================================================
# === 1️⃣ Données générales
# ================================================================

# --- Données physiques ---
ha_mat = 16.0       # hauteur du mât (m)
Ianc = 6.0          # distance d’ancrage (m)
Cd = 1.2            # coefficient de traînée du tube
dia_ext = 0.05      # diamètre extérieur du tube (m)
dia_int = 0.046     # diamètre intérieur du tube (m)
ent_tube = 0.29    # entraxe entre tubes (m)
A_lampe = 2.0       # surface de la lampe (m²)
m_lampe = 400       # masse de lampe kg
m_mat = 25 * 4      # masse du mat kg


# Constante
g = 9.81            # gravité terrestre
E = 70e9            # [Pa] (module d’élasticité de l’aluminium)
p_vent = 2700       # pression du vent (N/m²)

# Poids propre du mât (approximatif)
P_mat = m_mat * g  # N

# Force horizontale sur la lampe (vent sur lampe)
P_lampe = m_lampe * g

# charge verticale dans le mat du au mat et à la lampe (N)
Fec = P_mat + P_lampe

print(f"Poids du mat: {P_mat} N")
print(f"Poids de la lampe: {P_lampe} N")
print(f"Poids total {Fec} N")
```

```
Poids du mat: 981.0 N
Poids de la lampe: 3924.0 N
Poids total 4905.0 N
```

In [3]:

```python
# ================================================================
# === 2️⃣ Forces du vent
# ================================================================
Fvent_lampe = p_vent * A_lampe
Fvent_mat = p_vent * dia_ext * ha_mat * Cd
Fvent_total = Fvent_lampe + Fvent_mat

# Moment du vent autour du pied du mât
M_vent = Fvent_lampe * ha_mat + Fvent_mat * 0.6 * ha_mat

# Répartition du vent entre les deux niveaux
Fhaut = (2 * M_vent / ha_mat - Fvent_total)
Fbas  = Fvent_total - Fhaut
Fhaut_ratio = Fhaut / Fvent_total
Fbas_ratio  = Fbas / Fvent_total


# ================================================================
# === 3️⃣ Angles des haubans
# ================================================================
```

```python
alpha1 = sp.atan(ha_mat / Ianc)
alpha2 = sp.atan((ha_mat / 2) / Ianc)


# ==================================================================
# === 4️⃣ Tensions dans les haubans
# Hypothèse : T1,i = 2 * Ta,i
# ==================================================================
Ta1 = Fhaut / (3 * sp.cos(alpha1))
T11 = 2 * Ta1
Ta2 = Fbas / (3 * sp.cos(alpha2))
T12 = 2 * Ta2

Tsup = float(T11)
Tinf = float(T12)


# ==================================================================
# === 5️⃣ Réactions au pied du mât
# ==================================================================
Rx = -0.5 * sp.cos(alpha2) * (T12 + Ta2)
Ry = -sp.sqrt(3)/2 * (sp.cos(alpha1)*Ta1 + sp.cos(alpha2)*Ta2)
Rz = Fec + 3 * (T11 * sp.sin(alpha1) + T12 * sp.sin(alpha2))


# ==================================================================
# === 6️⃣ Décomposition 3D des haubans
# ==================================================================
def hauban_components(t, alpha, theta_deg):
    """Retourne (Tx, Ty, Tz) pour une tension T, un angle alpha et une orientati
    theta = np.deg2rad(theta_deg)
    tx = t * np.cos(alpha) * np.cos(theta)
    ty = t * np.cos(alpha) * np.sin(theta)
    tz = -t * np.sin(alpha)
    return np.array([tx, ty, tz])

# Haubans supérieurs (niveau 1)
H11 = hauban_components(float(T11), float(alpha1), 0)
HA1g = hauban_components(float(Ta1), float(alpha1), 120)
HA1d = hauban_components(float(Ta1), float(alpha1), 240)

# Haubans inférieurs (niveau 2)
H12 = hauban_components(float(T12), float(alpha2), 0)
HA2g = hauban_components(float(Ta2), float(alpha2), 120)
HA2d = hauban_components(float(Ta2), float(alpha2), 240)

# --- Résultantes par niveau ---
Sum_Tx1 = H11[0] + HA1g[0] + HA1d[0]
Sum_Ty1 = H11[1] + HA1g[1] + HA1d[1]
Sum_Tz1 = H11[2] + HA1g[2] + HA1d[2]

Sum_Tx2 = H12[0] + HA2g[0] + HA2d[0]
Sum_Ty2 = H12[1] + HA2g[1] + HA2d[1]
Sum_Tz2 = H12[2] + HA2g[2] + HA2d[2]

# --- Affichage principal ---
print("=== RÉSULTATS MISE À JOUR ===\n")
vals = {
    "α1 (°)"                : float(sp.deg(alpha1)),
    "α2 (°)"                : float(sp.deg(alpha2)),
```

```
    "Fvent_total (N)"      : float(Fvent_total),
    "Fhaut (N)"            : float(Fhaut),
    "Fbas (N)"             : float(Fbas),
    "Fhaut_ratio (%)"      : float(Fhaut_ratio * 100),
    "Fbas_ratio (%)"       : float(Fbas_ratio * 100),
    "T11 (N)"              : float(T11),
    "Ta1 (N)"              : float(Ta1),
    "T12 (N)"              : float(T12),
    "Ta2 (N)"              : float(Ta2),
    "Rx (N)"               : float(Rx),
    "Ry (N)"               : float(Ry),
    "Rz (N)"               : float(Rz),
}
for k, v in vals.items():
    print(f"{k:<25s} : {v:10.2f}")
```

```
=== RÉSULTATS MISE À JOUR ===

α1 (°)                     :      69.44
α2 (°)                     :      53.13
Fvent_total (N)            :    7992.00
Fhaut (N)                  :    5918.40
Fbas (N)                   :    2073.60
Fhaut_ratio (%)            :      74.05
Fbas_ratio (%)             :      25.95
T11 (N)                    :   11237.07
Ta1 (N)                    :    5618.54
T12 (N)                    :    2304.00
Ta2 (N)                    :    1152.00
Rx (N)                     :   -1036.80
Ry (N)                     :   -2307.09
Rz (N)                     :   41999.40
```

In [4]:

```
# ================================================================
# === 1🔟 Charge critique de flambage (Euler, haubans pris en compte)
# ================================================================

Ic = (np.pi / 64) * ((dia_ext**4) - (dia_int**4))
Atube = (np.pi / 4) * ((dia_ext**2) - (dia_int**2))
Itotal = 2 * Ic + 2 * (Ic + (Atube * ((ent_tube/2)**2)))

# --- Longueurs et coefficients de flambement ---
L1 = ha_mat / 2     # partie inférieure (bas-haubans)
L2 = ha_mat / 2     # partie supérieure (haut-haubans)
K1 = 0.7            # encastré - articulé
K2 = 1.0            # articulé - articulé

# --- Charges critiques partielles ---
Fcr1 = (np.pi**2 * E * Itotal) / ((K1 * L1)**2)
Fcr2 = (np.pi**2 * E * Itotal) / ((K2 * L2)**2)

# --- Charge critique équivalente (tronçons en série) ---
Fcr_eq = (Fcr1 * Fcr2) / (Fcr1 + Fcr2)

# --- Effort réel de compression dans le mât ---
effort_compression_montant = float(Rz)
Csecu = Fcr_eq / effort_compression_montant

# --- Affichage console ---
print("\n=== 🔩 Flambage d’Euler corrigé (avec haubans) ===")
```

```
print(f"Module E : {E:.2e} Pa")
print(f"Moment d’inertie total : {Itotal:.4e} m⁴")
print(f"Fcr bas (encastré-articulé) : {Fcr1:,.2f} N")
print(f"Fcr haut (articulé-articulé): {Fcr2:,.2f} N")
print(f"Fcr équivalent global       : {Fcr_eq:,.2f} N")
print(f"Effort compression réel      : {effort_compression_montant:,.2f} N")
print(f"Coefficient de sécurité      : {Csecu:.2f}")
```

```
=== 🔩 Flambage d’Euler corrigé (avec haubans) ===
Module E : 7.00e+10 Pa
Moment d’inertie total : 1.3030e-05 m⁴
Fcr bas (encastré-articulé) : 287,056.11 N
Fcr haut (articulé-articulé): 140,657.49 N
Fcr équivalent global       : 94,401.00 N
Effort compression réel      : 41,999.40 N
Coefficient de sécurité      : 2.25
```

In [5]:

```
# === 8️⃣ Visualisation 2D ===

# === Points principaux ===
origin = np.array([0, 0])
top = np.array([0, ha_mat])
mid = np.array([0, ha_mat / 2])
anchors = np.array([[-Ianc, 0], [Ianc, 0]])

# === Figure ===
fig, ax = plt.subplots(figsize=(12, 12))
ax.set_aspect('equal')
ax.set_title("Vue latérale d’un mât haubané à deux niveaux", fontsize=14, fontwe

# === Mât ===
ax.plot([origin[0], top[0]], [origin[1], top[1]], 'k-', lw=3, label="Mât")

# === Haubans ===
# === Haubans gauche uniquement ===
anchor_gauche = anchors[0]
ax.plot([top[0], anchor_gauche[0]], [top[1], anchor_gauche[1]], 'r-', lw=3, labe
ax.plot([mid[0], anchor_gauche[0]], [mid[1], anchor_gauche[1]], 'b--', lw=3, lab


# === Sol ===
ax.axhline(0, color='sienna', lw=3)
ax.text(-Ianc * 1.1, -0.8, f"Ancrage = ±{Ianc:.1f} m", color='sienna', fontsize=

# === Flèches de forces ===
scale = 0.00015
head = dict(head_width=0.7, head_length=0.7)

# --- Poids lampe ---
ax.arrow(top[0], top[1], 0, -P_lampe * scale * 20, color="brown", lw=3, **head)
ax.text(top[0] + 0.8, top[1] - 1.5, f"P_lampe = {P_lampe:.0f} N ↓", color="brown

# --- Vent lampe ---
ax.arrow(top[0], top[1], Fvent_lampe * scale * 1.5, 0, color="orange", lw=3, **h
ax.text(top[0] + Fvent_lampe * scale * 1.8, top[1] + 1.0,
        f"F_vent lampe = {Fvent_lampe:.0f} N →", color="orange", fontsize=10)

# --- Poids mât ---
ax.arrow(mid[0], mid[1], 0, -P_mat * scale * 3, color="brown", lw=3, **head)
ax.text(mid[0] + 0.8, mid[1] - 1.0, f"P_mât = {P_mat:.0f} N ↓", color="brown", f
```

```python
# --- Vent mât ---
ax.arrow(mid[0], mid[1], Fvent_mat * scale * 1.5, 0, color="darkorange", lw=3, *
ax.text(mid[0] + Fvent_mat * scale * 1.8, mid[1] + 0.6,
        f"F_vent mât = {Fvent_mat:.0f} N →", color="darkorange", fontsize=10)

# --- Réactions sol ---
ax.arrow(0, 0.25 * ha_mat, float(Rx) * scale, 0, color="darkred", lw=3, **head)
ax.text(0.3, 0.25 * ha_mat + 0.5, f"Rₓ = {float(Rx):.0f} N", color="darkred", fc

ax.arrow(0, 0.15 * ha_mat, 0, float(Rz) * scale, color="green", lw=3, **head)
ax.text(0.3, 0.15 * ha_mat + 0.5, f"Rz = {float(Rz):.0f} N", color="green", font

# --- Tensions haubans (côté gauche uniquement) ---
for anchor, couleur, T, niveau, z0, dx_text, dy_text in [
    (anchors[0], "purple", Tsup, "sup", ha_mat, -1.0, 0.8),
    (anchors[0], "teal",   Tinf, "inf", ha_mat/2, -1.0, -0.8),
]:
    dx = anchor[0] - 0
    dz = anchor[1] - z0
    L = np.hypot(dx, dz)
    ux, uz = dx / L, dz / L
    ax.arrow(0, z0, ux * T * scale, uz * T * scale, color=couleur, lw=4, alpha=0
    txt_x = ux * T * scale * 1.1 + dx_text
    txt_y = z0 + uz * T * scale * 1.1 + dy_text
    ax.text(txt_x, txt_y, f"T_{niveau} = {T:.0f} N", color=couleur,
            fontsize=10, ha='center', va='center',
            bbox=dict(facecolor='white', alpha=0.85, edgecolor='none', pad=1))

# === Lampe ===
ax.plot(top[0], top[1], 'yo', ms=12)
ax.text(top[0] + 0.3, top[1] + 0.5, "Lampe", color='orange', fontsize=10, fontwe

# === Résumé texte (déplacé à droite) ===
def format_force(val):
    return f"{val/1000:.2f} kN" if abs(val) >= 1000 else f"{val:.0f} N"

text_x = Ianc * 1.2
text_y = ha_mat * 1.1

info = (
    f"α₁ = {float(sp.deg(alpha1)):.1f}°\n"
    f"α₂ = {float(sp.deg(alpha2)):.1f}°\n"
    f"T_sup = {format_force(Tsup)}\n"
    f"T_inf = {format_force(Tinf)}\n"
    f"Rₓ = {format_force(float(Rx))}\n"
    f"Rz = {format_force(float(Rz))}\n"
    f"----------------------\n"
    f"⚙️ Stabilité du mât :\n"
    f"Fcr₁ = {format_force(Fcr1)} (bas)\n"
    f"Fcr₂ = {format_force(Fcr2)} (haut)\n"
    f"Fcr(eq) = {format_force(Fcr_eq)}\n"
    f"Charge compression = {format_force(effort_compression_montant)}\n"
    f"Csécurité = {Csecu:.2f}"
)
ax.text(text_x, text_y, info, fontsize=10, color="black",
        bbox=dict(facecolor="white", alpha=0.8, boxstyle="round,pad=0.5"))

# === Axes & sortie ===
ax.set_xlim(-Ianc * 2.2, Ianc * 2.2)
```

```
ax.set_ylim(-3, ha_mat * 1.6)
ax.set_xlabel("x (m)")
ax.set_ylabel("z (m)")
ax.legend(loc="upper right")
ax.grid(True, linestyle="--", alpha=0.4)
plt.tight_layout()

plt.savefig("mat_haubane.png", dpi=300)
plt.show()
```