AugerBot Calculations

```
Quit
```

Trial 1: 9/19 - 9/26

Modified Francisco Calculations : 10/18 (Do not use!)

Plotting Fx to find U which Balances Forces: 11/1

Plotting Fx to find U which Balances Forces 2 : 11/9

Inside Equation for Thrust: 11/16

Backtracking: 11/27

- Dimensionalized Fx to Nondimensionalized Fx ✓

Francisco' s Eqn w/ Constant Ct, Cn, and N (turns)

In[37]:=
```
Quit;
```

Setup

In[1]:=
```
(*Parameters from paper*)
r = 5/1000; w = 10.4; n = 3.5;
c = 24; d = 3/1000; ρ = 1.45*10^3; g = 9.8; h = 50/1000;
Um = 0.11; ϕ0 = 16*Pi/180;
```

```
In[4]:= (*Solving for friction coefficients used in Figure 3 of Francisco's paper*)
eq1 = (CN / CT) - ((1 + Um * Tan[ϕ0]) / (1 - Um / Tan[ϕ0]));
eq2 = CN - CT - c * d * ρ * g * h;
coeffSol = NSolve[{eq1 == 0, eq2 == 0}, {CT, CN}][[1]];
Ct = coeffSol[[1, 2]]
Cn = coeffSol[[2, 2]]

Out[7]= 75.9513

Out[8]= 127.107

In[9]:= (*Eq 2 - 10/21/17 | Parametrized f Integrated only wrt dθ*)
Fran[u_] := (2 * Pi * n * r / Cos[ϕ]) *
   ((Cn - Ct) * r * w * Sin[ϕ] * Cos[ϕ] - u * (Ct * Sin[ϕ]^2 + Cn * Cos[ϕ]^2)) /
    Sqrt[u^2 + (r * w)^2];
```

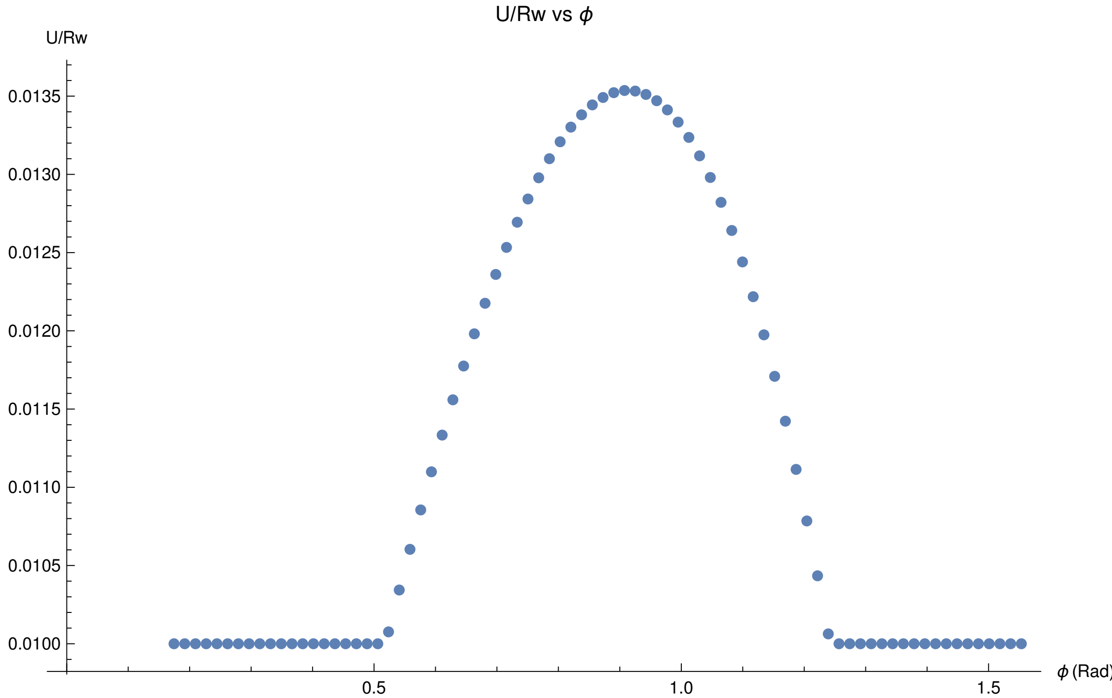
Calculating U/Rw when Fx = 0 for Many ϕ Cases

```
In[10]:= ϕ = 10 * Pi / 180 // N; (*Local inclination, radians*)
ϕstore = {};
ustore = {};
Fstore = {};
FMaxstore = {}; (*FxIn when u = 0.001*)

In[14]:= (*Finding U/Rw intercepts*)
While[ϕ < 90 * Pi / 180,
 (*Print statements*)
 (*Print["Let ϕ = ",ϕ*180/Pi, " deg"];*)
 (*Print@Plot[{Fran[u],0},{u,0,0.2}, PlotLabel→"Inner Fx vs. Helix Velocity U/Rw",
   AxesLabel→{"U/Rw (m/s)","Fx (N/m^2)"}, PlotRange→All];*)

 (*Finding U/Rw intercept: Newton-Raphson Method*)
 guess = 0.01; (*Reset initial guess*)
 grad = D[Fran[u], u];
 While[Fran[guess] > 10^-6, (*Keep iterating until FxIn ≃ 0*)
  gradEval = grad /. u → guess; (*Find FxIn'(guess)*)
  guess = guess - Fran[guess] / gradEval (*u_i+1 = u_i - FxIn(u_i)/FxIn'(u_i)*)
 ];
 uint = guess; (*U/Rw intercept found*)

 (*Storing data in arrays*)
 ϕstore = Join[ϕstore, {ϕ}]; (*Storing ϕ in Radians*)
 FMaxstore = Join[FMaxstore, {Fran[10^-3]}]; (*FxIn(0.001)*)
 ustore = Join[ustore, {uint}]; (*U/Rw found when FxIn < 10^-6*)
 Fstore = Join[Fstore, {Fran[uint]}]; (*FxIn val at u-intercept*)

 ϕ = ϕ + (1 * Pi / 180) (*Increment by 1 deg*)
]
```

Plotting Data

In[19]:=
```
ϕFcol = {ϕstore}ᵀ; uFcol = {ustore}ᵀ;

(*Helix Translation Speed vs. Local Incline Angle ϕ*)
dataPlot = Join[ϕFcol, uFcol, 2];
ListPlot[dataPlot, PlotLabel → "U/Rw vs ϕ",
 AxesLabel → {"ϕ (Rad)", "U/Rw"}, ImageSize → Large]
```

Out[21]=



In[18]:=

When friction coefficients Ct and Cn are held constant, a maximum helix translational speed does occur. A significant increase in U/Rw starts around 28.6479 degrees.
Note: the number of turns (n) was kept the same. This means that the projected length of the helix along the x-axis is not constant.

Francisco With Chen' s Coefficients

In[22]:=
```
Quit;
```

In[1]:=
```
(*Parameters from paper*)
r = 5/1000; w = 10.4; n = 3.5;

zlpGlass = 0.3; xlpGlass = 0.075;
β = (Pi/2) - ϕ; (*ϕ is symbolic, radians*)
αz = zlpGlass * Abs[Cos[β]] * (100^3); (*Vertical stress per unit depth, N/m^3*)
αx = xlpGlass * Abs[Sin[β]] * (100^3); (*Horizontal stress per unit depth, N/m^3*)

d = 0.05; (*Depth robot buried, 50mm*)

(*Friction coefficients, expressed in terms of ϕ*)
Cn = αz * d; (*N/m^2*)
Ct = αx * d;

(*Eq 2 - 10/21/17 | Parametrized f Integrated only wrt dθ*)
FranChen[u_] := (2 * Pi * n * r / Cos[ϕ]) *
   ((Cn - Ct) * r * w * Sin[ϕ] * Cos[ϕ] - u * (Ct * Sin[ϕ]^2 + Cn * Cos[ϕ]^2)) /
    Sqrt[u^2 + (r * w)^2];
```

Calculating U/Rw when Fx = 0 for Many ϕ Cases

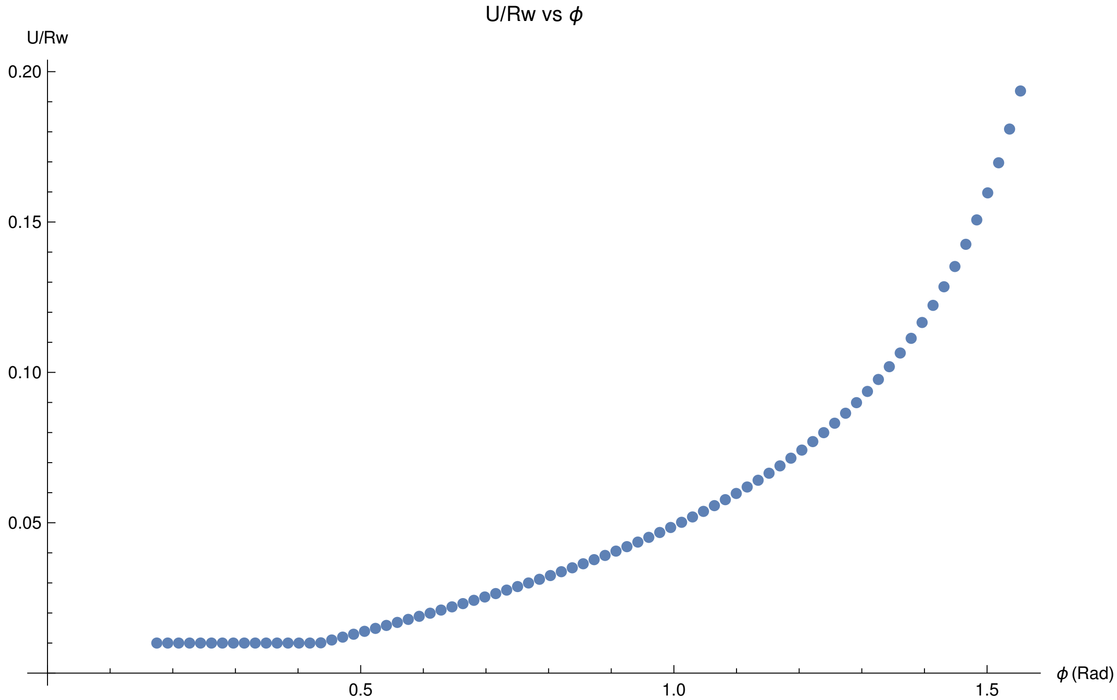
In[12]:=
```
ϕ = 10 * Pi/180 // N; (*Local inclination, radians*)
ϕstore = {};
ustore = {};
Fstore = {};
FMaxstore = {}; (*FxIn when u = 0.001*)
```

In[16]:=
```
(*Finding U/Rw intercepts*)
While[ϕ < 90 * Pi / 180,
 (*Print statements*)
 (*Print["Let ϕ = ",ϕ*180/Pi, " deg"];*)
 (*Print@
   Plot[{FranChen[u],0},{u,0,0.2}, PlotLabel→"Inner Fx vs. Helix Velocity U/Rw",
    AxesLabel→{"U/Rw (m/s)","Fx (N/m^2)"}, PlotRange→All];*)

 (*Finding U/Rw intercept: Newton-Raphson Method*)
 guess = 0.01; (*Reset initial guess*)
 grad = D[FranChen[u], u];
 While[FranChen[guess] > 10^-6, (*Keep iterating until FxIn ≃ 0*)
  gradEval = grad /. u → guess; (*Find FxIn'(guess)*)
  guess = guess - FranChen[guess] / gradEval (*u_{i+1} = u_i - FxIn(u_i)/FxIn'(u_i)*)
 ];
 uint = guess; (*U/Rw intercept found*)

 (*Storing data in arrays*)
 ϕstore = Join[ϕstore, {ϕ}]; (*Storing ϕ in Radians*)
 FMaxstore = Join[FMaxstore, {FranChen[10^-3]}]; (*FxIn(0.001)*)
 ustore = Join[ustore, {uint}]; (*U/Rw found when FxIn < 10^-6*)
 Fstore = Join[Fstore, {FranChen[uint]}]; (*FxIn val at u-intercept*)

 ϕ = ϕ + (1 * Pi / 180) (*Increment by 1 deg*)
]
```

Plotting Francisco - Chen Results

In[17]:=
```
ϕFCcol = {ϕstore}ᵀ; uFCcol = {ustore}ᵀ;

(*Helix Translation Speed vs. Local Incline Angle ϕ*)
dataPlot = Join[ϕFCcol, uFCcol, 2];
ListPlot[dataPlot, PlotLabel → "U/Rw vs ϕ",
 AxesLabel → {"ϕ (Rad)", "U/Rw"}, ImageSize → Large]
```

Out[19]=



Evidently, Chen' s variable Ct and Cn coefficients mess with Francisco' s equation in a way which does not intuitively make sense.