Frenet-Serret / Lagrange Polynomial (FSLP) Model

Notebook example for the FSLP flux rope model. ...

In[∘]:=
```
RFrenetSerretEqs = {
    γ′[s] → t[s] × l[s],
    t′[s]  → κ[s] n[s] × l[s],
    n′[s] → -κ[s] t[s] × l[s] + τ[s] b[s] × l[s],
    b′[s] → -τ[s] n[s] × l[s]
  };

RFrenetSerretGeometry = {
    t[s]⊗t[s]  → 1,
    n[s]⊗n[s]  → 1,
    b[s]⊗b[s]  → 1,

    t[s]⊗n[s]  → 0,
    n[s]⊗t[s]  → 0,
    t[s]⊗b[s]  → 0,
    b[s]⊗t[s]  → 0,
    n[s]⊗b[s]  → 0,
    b[s]⊗n[s]  → 0,

    t[s]×t[s]  → 0,
    n[s]×n[s]  → 0,
    b[s]×b[s]  → 0,

    t[s]×n[s]  → b[s],
    n[s]×b[s]  → t[s],
    b[s]×t[s]  → n[s],
    n[s]×t[s]  → -b[s],
    t[s]×b[s]  → -n[s],
    b[s]×n[s]  → -t[s]
  };

RFrenetSerretVariables = {
    r ∈ PositiveReals,
    σ ∈ PositiveReals,
    l[s] ∈ PositiveReals,
    κ[s]  ∈ PositiveReals,
    τ[s] ∈ Reals,
```

```
   γ[s] ∈ Vectors[3, Reals],
   t[s] ∈ Vectors[3, Reals],
   t′[s] ∈ Vectors[3, Reals],
   n[s] ∈ Vectors[3, Reals],
   n′[s] ∈ Vectors[3, Reals],
   b′[s] ∈ Vectors[3, Reals],
   b′[s] ∈ Vectors[3, Reals]
  };

RLegendreRulesr0 = {
   (2 ((-1 - i_) (-1 + 2 r) LegendreP[i_, -1 + 2 r] + (1 + i_) LegendreP[1 + i_, -1 + 2 r])) / (-1 + (-1 + 2 r)^2) :→
    (-1)^(i+1) (i^2 + i)
  };

FSv[r_, s_, φ_] := γ[s] - r σ n[s] × Cos[φ] - r σ b[s] × Sin[φ];

FSϵr[r_, s_, φ_] := D[FSv[r, s, φ], r] /. RFrenetSerretEqs
FSϵs[r_, s_, φ_] := D[FSv[r, s, φ], s] /. RFrenetSerretEqs // FullSimplify
FSϵφ[r_, s_, φ_] := D[FSv[r, s, φ], φ] /. RFrenetSerretEqs // FullSimplify
FSgrr[r_, s_, φ_] := TensorExpand[FSϵr[r, s, φ]⊗FSϵr[r, s, φ] , Assumptions →
     RFrenetSerretVariables] /. RFrenetSerretGeometry // FullSimplify
FSgss[r_, s_, φ_] := TensorExpand[FSϵs[r, s, φ]⊗FSϵs[r, s, φ] , Assumptions →
     RFrenetSerretVariables] /. RFrenetSerretGeometry // FullSimplify
FSgφφ[r_, s_, φ_] := TensorExpand[FSϵφ[r, s, φ]⊗FSϵφ[r, s, φ] , Assumptions →
     RFrenetSerretVariables] /. RFrenetSerretGeometry // FullSimplify
FSgrs[r_, s_, φ_] := TensorExpand[FSϵr[r, s, φ]⊗FSϵs[r, s, φ] , Assumptions →
     RFrenetSerretVariables] /. RFrenetSerretGeometry // FullSimplify
FSgrφ[r_, s_, φ_] := TensorExpand[FSϵr[r, s, φ]⊗FSϵφ[r, s, φ] , Assumptions →
     RFrenetSerretVariables] /. RFrenetSerretGeometry // FullSimplify
FSgsφ[r_, s_, φ_] := TensorExpand[FSϵs[r, s, φ]⊗FSϵφ[r, s, φ] , Assumptions →
     RFrenetSerretVariables] /. RFrenetSerretGeometry // FullSimplify
FSgij[r_, s_, φ_] := {{FSgrr[r, s, φ], FSgrs[r, s, φ], FSgrφ[r, s, φ]},
   {FSgrs[r, s, φ], FSgss[r, s, φ], FSgsφ[r, s, φ]},
   {FSgrφ[r, s, φ], FSgsφ[r, s, φ], FSgφφ[r, s, φ]}};
FSsqg[r_, s_, φ_] :=
 FullSimplify[Sqrt[Det[FSgij[r, s, φ]] // FullSimplify], Assumptions →
     RFrenetSerretVariables] /. Sqrt[x_^2 y_^2] :→ x y /. Sqrt[x_^2 ] :→ x

FShr[r_, s_, φ_] :=
 FullSimplify[Sqrt[FSgrr[r, s, φ]], Assumptions → RFrenetSerretVariables]
FShs[r_, s_, φ_] :=
 FullSimplify[Sqrt[FSgss[r, s, φ]], Assumptions → RFrenetSerretVariables]
FShφ[r_, s_, φ_] :=
```

```
FullSimplify[Sqrt[FSg_φφ[r, s, φ]], Assumptions → RFrenetSerretVariables]

FSB_r[r_, s_, φ_] := 0
FSB_s[r_, s_, φ_] :=
```

$$-\frac{1}{l[s]}\left(\frac{\mu_0\,\sigma\,\sqrt{1-r^2\,\sigma^2\,\kappa[s]^2}}{(1+r\,\sigma\,\mathrm{Cos}[\varphi]\,\kappa[s])^2}+\frac{\mu_0\,\sigma\,\hat{A}_1[r,s,\varphi]}{1+r\,\sigma\,\mathrm{Cos}[\varphi]\,\kappa[s]}\right)\sum_{n=0}^{n_m}\alpha[n]\,\mathrm{LegendreP}[n,\,2\,r-1]$$

$$\mathrm{FSB}_\varphi[r\_,s\_,\varphi\_]:=\frac{-\mu_0}{(1+r\,\sigma\,\mathrm{Cos}[\varphi]\,\kappa[s])}\left(\sum_{m=0}^{m_m}\beta[m]\,\mathrm{LegendreP}[m,\,2\,r-1]\right)+$$

$$\frac{\mu_0\,\sigma\left(-\frac{r\,\sigma\,\mathrm{Sin}[\varphi]\,\kappa'[s]}{\sqrt{1-r^2\,\sigma^2\,\kappa[s]^2}}+\mathrm{Integrate}\left[\mathrm{D}\left[\hat{A}_1[r,s,\varphi_t],s\right],\{\varphi_t,0,\varphi\}\right](1+r\,\sigma\,\mathrm{Cos}[\varphi]\,\kappa[s])\right)}{l[s]\,(1+r\,\sigma\,\mathrm{Cos}[\varphi]\,\kappa[s])^2}$$

$$\sum_{n=0}^{n_m}\alpha[n]\,\mathrm{LegendreP}[n,\,2\,r-1]$$

```
FSB_D[r_, s_, φ_] :=
 D[FSsqg[r, s, φ] FSB_s[r, s, φ], s] + D[FSsqg[r, s, φ] FSB_φ[r, s, φ], φ]
FSJ_r[r_, s_, φ_] :=
 (D[FSg_sφ[r, s, φ] FSB_s[r, s, φ] + FSg_φφ[r, s, φ] FSB_φ[r, s, φ], s] - D[FSg_ss[r, s, φ]
       FSB_s[r, s, φ] + FSg_sφ[r, s, φ] FSB_φ[r, s, φ], φ]) / FSsqg[r, s, φ] / μ0
FSJ_s[r_, s_, φ_] :=
 (D[FSg_rs[r, s, φ] FSB_s[r, s, φ] + FSg_rφ[r, s, φ] FSB_φ[r, s, φ], φ] - D[FSg_sφ[r, s, φ]
       FSB_s[r, s, φ] + FSg_φφ[r, s, φ] FSB_φ[r, s, φ], r]) / FSsqg[r, s, φ] / μ0
FSJ_φ[r_, s_, φ_] :=
 (D[FSg_ss[r, s, φ] FSB_s[r, s, φ] + FSg_sφ[r, s, φ] FSB_φ[r, s, φ], r] - D[FSg_rs[r, s, φ]
       FSB_s[r, s, φ] + FSg_rφ[r, s, φ] FSB_φ[r, s, φ], s]) / FSsqg[r, s, φ] / μ0
FSJ_D[r_, s_, φ_] := D[FSsqg[r, s, φ] FSJ_r[r, s, φ], r] +
  D[FSsqg[r, s, φ] FSJ_s[r, s, φ], s] + D[FSsqg[r, s, φ] FSJ_φ[r, s, φ], φ]

FSF_r[r_, s_, φ_] := Simplify[Inverse[FSg_ij[r, s, φ]]] [[1]] [[1]] ×
  FSsqg[r, s, φ] (FSJ_s[r, s, φ] FSB_φ[r, s, φ] - FSJ_φ[r, s, φ] FSB_s[r, s, φ])
FSF_s[r_, s_, φ_] := Simplify[Inverse[FSg_ij[r, s, φ]]] [[2]] [[2]] × FSsqg[r, s, φ]
   (-FSJ_r[r, s, φ] FSB_φ[r, s, φ]) + Simplify[Inverse[FSg_ij[r, s, φ]]] [[2]] [[3]] ×
   FSsqg[r, s, φ] (FSJ_r[r, s, φ] FSB_s[r, s, φ])
FSF_φ[r_, s_, φ_] := Simplify[Inverse[FSg_ij[r, s, φ]]] [[3]] [[3]] × FSsqg[r, s, φ]
   (FSJ_r[r, s, φ] FSB_s[r, s, φ]) + Simplify[Inverse[FSg_ij[r, s, φ]]] [[2]] [[3]] ×
   FSsqg[r, s, φ] (-FSJ_r[r, s, φ] FSB_φ[r, s, φ])

SetGeometryCylinder = {κ[s] → 0, κ'[s] → 0, κ''[s] → 0, τ[s] → 0, τ'[s] → 0,
   τ''[s] → 0, x_^(0,1,0)[a_, b_, c_] :→ 0, x_^(0,2,0)[a_, b_, c_] :→ 0};
SetGeometryTorus := {κ'[s] → 0, κ''[s] → 0, τ[s] → 0, τ'[s] → 0,
   τ''[s] → 0, x_^(0,1,0)[a_, b_, c_] :→ 0, x_^(0,2,0)[a_, b_, c_] :→ 0};
SetZeroAFunc := {Â1[r, s, p_] :→ 0, Â1^(1,0,0)[r, s, p_] :→ 0,
```

```
Â1^(0,1,0)[r, s, p_] :> 0, Â1^(0,0,1)[r, s, p_] :> 0, Â1^(2,0,0)[r, s, p_] :> 0,
Â1^(0,2,0)[r, s, p_] :> 0, Â1^(0,0,2)[r, s, p_] :> 0 , Â1^(1,1,0)[r, s, p_] :> 0,
Â1^(1,2,0)[r, s, y_] :> 0, Â1^(1,0,1)[r, s, y_] :> 0}
SetAFuncTo[F_] := {Â1[r, s, p_] :> F[r, s, p],
  Â1^(1,0,0)[r, s, p_] :> D[F[r, s, p], r], Â1^(0,1,0)[r, s, p_] :> D[F[r, s, p], s],
  Â1^(0,0,1)[r, s, p_] :> D[F[r, s, p], p], Â1^(2,0,0)[r, s, p_] :> D[F[r, s, p], {r, 2}],
  Â1^(0,2,0)[r, s, p_] :> D[F[r, s, p], {s, 2}],
  Â1^(0,0,2)[r, s, p_] :> D[F[r, s, p], {p, 2}],
  Â1^(1,1,0)[r, s, p_] :> D[D[F[r, s, p], s], r], Â1^(1,2,0)[r, s, y_] :>
   D[D[F[r, s, p], {s, 2}], r], Â1^(1,0,1)[r, s, y_] :> D[D[F[r, s, p], p], r]}

Bv[r_, s_, φ_] := FSϵs[r, s, φ] FSBs[r, s, φ] + FSϵφ[r, s, φ] FSBφ[r, s, φ]
Jv[r_, s_, φ_] :=
 FSϵr[r, s, φ] FSJr[r, s, φ] + FSϵs[r, s, φ] FSJs[r, s, φ] + FSϵφ[r, s, φ] FSJφ[r, s, φ]
Fv[r_, s_, φ_] :=
 FSϵr[r, s, φ] FSFr[r, s, φ] + FSϵs[r, s, φ] FSFs[r, s, φ] + FSϵφ[r, s, φ] FSFφ[r, s, φ]
```

In[*]:=

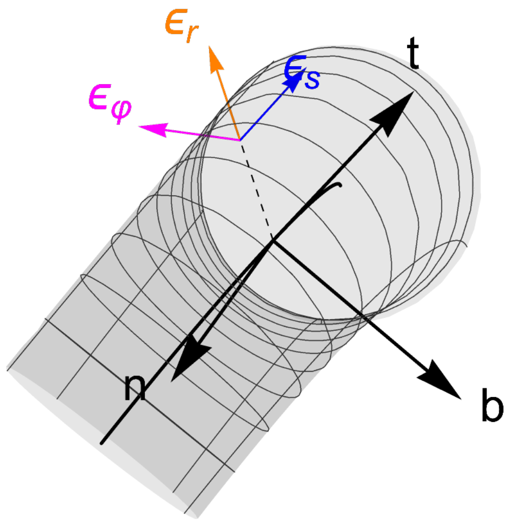
```
LPCoeff[n_] := Integrate[B0/(1 + γ0^2 r^2) LegendreP[n, 2 r - 1] (2 n + 1),
   {r, 0, 1}, Assumptions → {γ0 ∈ Reals}] // Normal
ConfigGH = {
   β[m_] :> γ0 α[m],
   nm → 12, mm → 12,
   α[0] → -LPCoeff[0],
   α[1] → -LPCoeff[1],
   α[2] → -LPCoeff[2],
   α[3] → -LPCoeff[3],
   α[4] → -LPCoeff[4],
   α[5] → -LPCoeff[5],
   α[6] → -LPCoeff[6],
   α[7] → -LPCoeff[7],
   α[8] → -LPCoeff[8],
   α[9] → -LPCoeff[9],
   α[10] → -LPCoeff[10],
   α[11] → -LPCoeff[11]
  };
ConfigGH = Join[ConfigGH,
   Solve[Sum[(-1)^(1+n) (n + n^2) α[n], {n, 0, nm}] == 0 //. ConfigGH, α[12]] // Flatten];
```

$$\mathrm{Solve}\left[\sum_{n=0}^{n_m} (-1)^{1+n}\left(n+n^2\right)\alpha[n] == 0 \;//.\; \mathrm{ConfigGH}, \alpha[12]\right]$$

```
In[•]:= ConfigGeneral = {
    μ0 → 1,
    B0 → 15 / σ,
    γ0 → 2,
    σ → .1,
    r0 → 1
   };

Eγ[s_] := {(1 - s / 2) Sin[2 Pi s], (1 - s / 1) Cos[2 Pi s], (1 + Sin[Pi s / 2]^2) / 2 Pi}
EDγ[s_] := Norm[D[Eγ[s] // ComplexExpand, s]]
Et[s_] := D[Eγ[s] // ComplexExpand, s] // Normalize
En[s_] := D[Et[s] // ComplexExpand, s] // Normalize
Eb[s_] := Et[s] × En[s] // Normalize
Er[s_] := Eγ[s] - r σ En[s] × Cos[φ] - r σ Eb[s] × Sin[φ];

Eκ[s_] := Sqrt[Cross[Eγ'[s], Eγ''[s]].Cross[Eγ'[s], Eγ''[s]] / (Eγ'[s].Eγ'[s])^3]
Eτ[s_] := Eγ'[s].Cross[Eγ''[s], Eγ^(3)[s]] / Norm[Eγ'[s] × Eγ''[s]]^2

Eϵr[s_] := -σ Cos[φ] × En[s] - σ Eb[s] × Sin[φ]
Eϵs[s_] :=
 Et[s] + r σ En[s] × Sin[φ] × Eτ[s] + r σ Cos[φ] (Et[s] × Eκ[s] - Eb[s] × Eτ[s])
Eϵφ[s_] := r σ (-Eb[s] × Cos[φ] + En[s] × Sin[φ])

ConfigE = {γ[s] → Eγ[s], t[s] → Et[s], n[s] → En[s], b[s] → Eb[s],
   κ[s] → Eκ[s], D[κ[s], s] → D[Eκ[s] // ComplexExpand, s],
   D[κ[s], {s, 2}] → D[Eκ[s] // ComplexExpand, {s, 2}],
   τ[s] → Eτ[s], D[τ[s], s] → D[Eτ[s] // ComplexExpand, s],
   l[s] → Norm[D[Eγ[s] // ComplexExpand, s]],
   D[l[s], s] → D[Norm[D[Eγ[s] // ComplexExpand, s]] // ComplexExpand, s],
   t[s] → Et[s], n[s] → En[s], b[s] → Eb[s]};

In[•]:= EP = {u0 → 4 Pi / 5, φ0 → Pi / 4};
FSPlot = Show[
  {
   ParametricPlot3D[Eγ[u] // Evaluate,
    {u, 3.9 Pi / 5, 4.1 Pi / 5}, PlotStyle → {Black, Thickness[.005]}],

   ParametricPlot3D[Er[u] /. ConfigGeneral /. r → 1 // Evaluate,
    {u, 3.9 Pi / 5, 4.1 Pi / 5}, {φ, 0, 2 Pi},
    PlotStyle → {Opacity[0.1], Black}, Mesh → {10, 4}],
   Graphics3D[{Arrowheads[.05], Thickness[.005], Black,
     {Arrow[{Eγ[u], Eγ[u] + 2 σ Et[u]} /. u → u0 /. ConfigGeneral /. EP]}}],
   Graphics3D[{Arrowheads[.05], Thickness[.005], Black,
     {Arrow[{Eγ[u], Eγ[u] + 2 σ En[u]} /. u → u0 /. ConfigGeneral /. EP]}}],
```

```
  Graphics3D[{Arrowheads[.05], Thickness[.005], Black,
    {Arrow[{Eγ[u], Eγ[u] + 2 σ Eb[u]} /. u → u0 /. ConfigGeneral /. EP]}}],
  Graphics3D[
   {Black, Text[Style["t", 24], {Eγ[u] + 2.25 σ Et[u] - 0.2 σ Eb[u]} /. u → u0 /.
         φ → φ0 /. ConfigGeneral /. EP /. r → 1.25]}],
  Graphics3D[
   {Black, Text[Style["n", 24], {Eγ[u] + 2.25 σ En[u] - 0.2 σ Eb[u]} /. u → u0 /.
         φ → φ0 /. ConfigGeneral /. EP /. r → 1.25]}],
  Graphics3D[
   {Black, Text[Style["b", 24], {Eγ[u] + 2.25 σ Eb[u] - 0.2 σ En[u]} /. u → u0 /.
         φ → φ0 /. ConfigGeneral /. EP /. r → 1.25]}],

  ParametricPlot3D[Er[u] /. u → u0 /. φ → φ0 /. ConfigGeneral /. EP,
   {r, 0, 1}, PlotStyle → {Black, Thickness[.002], Dashed}],
  Graphics3D[{Arrowheads[.03], Thickness[.003], Orange,
    {Arrow[{Er[u], Er[u] + σ Normalize[Eϵr[u]]} /. u → u0 /. φ → φ0 /.
         ConfigGeneral /. EP /. r → 1]}}],
  Graphics3D[{Arrowheads[.03], Thickness[.003], Blue,
    {Arrow[{Er[u], Er[u] + σ Normalize[Eϵs[u]]} /. u → u0 /. φ → φ0 /.
         ConfigGeneral /. EP /. r → 1]}}],
  Graphics3D[{Arrowheads[.03], Thickness[.003], Magenta,
    {Arrow[{Er[u], Er[u] + σ Normalize[Eϵφ[u]]} /. u → u0 /. φ → φ0 /.
         ConfigGeneral /. EP /. r → 1]}}],
  Graphics3D[{Orange , Text[Style["ϵr", 24],
     {Er[u] + σ Normalize[Eϵr[u]] + 0.2 σ Normalize[Eϵφ[u]]} /. u → u0 /.
         φ → φ0 /. ConfigGeneral /. EP /. r → 1.25]}],
  Graphics3D[{Blue    , Text[Style["ϵs", 24],
     {Er[u] + 0.75 σ Normalize[Eϵs[u]] - 0.2 σ Normalize[Eϵφ[u]]} /. u → u0 /.
         φ → φ0 /. ConfigGeneral /. EP /. r → 1.25]}],
  Graphics3D[{Magenta, Text[Style["ϵφ", 24],
     {Er[u] + σ Normalize[Eϵφ[u]] + 0.2 σ Normalize[Eϵφ[u]]} /. u → u0 /.
         φ → φ0 /. ConfigGeneral /. EP /. r → 1.25]}]
 },
 PlotRange → All, Boxed → False, Axes → False
]
```

Out[ ]=

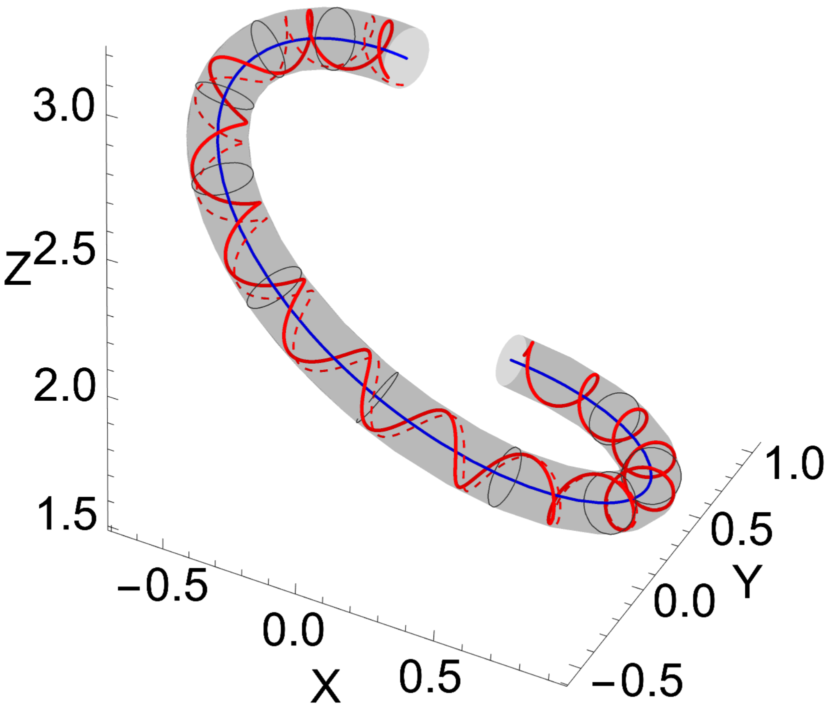


In[ ]:=

```
smax = 0;
p[t_] := {p1[t], p2[t], p3[t]}
RCoords = {r → p1[t], s → p2[t], φ → p3[t]};
eqsB = Thread[
   D[FSv[r, s, φ] //. ConfigE //. ConfigGeneral /. RCoords // ComplexExpand, t] ==
         Bv[r, s, φ] / Norm[Bv[r, s, φ]] /. SetZeroAFunc //.
       ConfigE //. ConfigGH //. ConfigGeneral /. RCoords];
eqsInitial := Thread[p[0] == {.9, 0, 0}];
eqsSolve = NDSolve[{eqsB, eqsInitial, WhenEvent[p2[t] ≥ 1, smax = t;
      "StopIntegration"]}, {p1, p2, p3}, {t, 0, Infinity},
    Method → {"EquationSimplification" → "Residual"}] // Flatten;

smaxNaive = 0;
eqsBNaive = Thread[
   D[FSv[r, s, φ] //. ConfigE //. ConfigGeneral /. RCoords // ComplexExpand, t] ==
          Bv[r, s, φ] / Norm[Bv[r, s, φ]] /. SetGeometryTorus /. SetZeroAFunc //.
       ConfigE //. ConfigGH //. ConfigGeneral /. RCoords];
eqsSolveNaive =
  NDSolve[{eqsBNaive, eqsInitial, WhenEvent[p2[t] ≥ 1, smaxNaive = t;
      "StopIntegration"]}, {p1, p2, p3}, {t, 0, Infinity},
    Method → {"EquationSimplification" → "Residual"}] // Flatten;
```

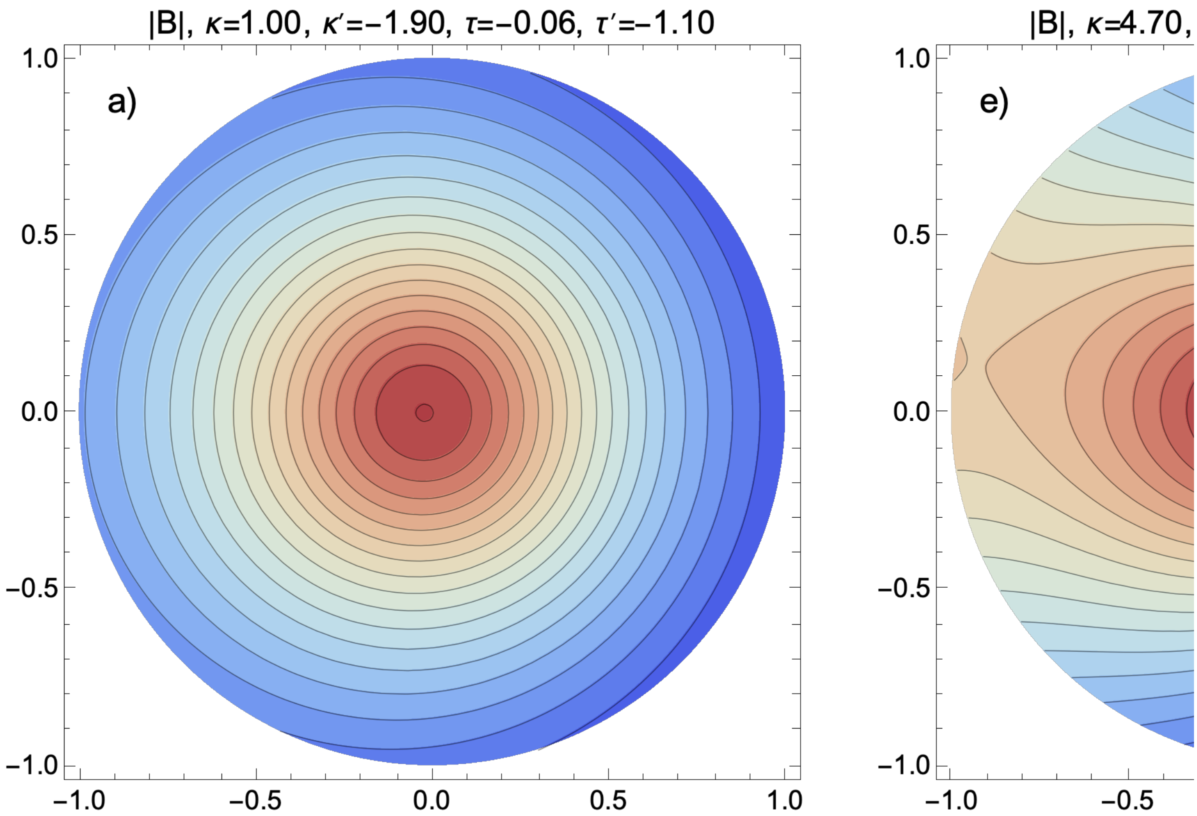
In[•]:=
```
Show[
 {
  ParametricPlot3D[Eγ[s] /. ConfigGeneral // Evaluate,
   {s, 0, 1}, PlotStyle → {Opacity[1], Blue, Thickness[.004]}],
  ParametricPlot3D[Er[s] /. ConfigGeneral /. r → 1 // Evaluate, {s, 0, 1},
   {φ, 0, 2 Pi }, PlotStyle → {Opacity[0.15], Black}, Mesh → {10, 0}],
  ParametricPlot3D[
   FSv[r, s, φ] //. ConfigE //. ConfigGeneral /. RCoords /. eqsSolve // Evaluate,
   {t, 0, smax}, PlotStyle → {Red, Thickness[.005]}],
  ParametricPlot3D[
   FSv[r, s, φ] //. ConfigE //. ConfigGeneral /. RCoords /. eqsSolveNaive //
    Evaluate, {t, 0, smaxNaive}, PlotStyle → {Red, Thickness[.003], Dashed}]
 },
 AxesLabel → {"X", "Y", "Z"}, Boxed → False, TicksStyle → Large,
 AxesStyle → Large, ViewPoint → {2, -4, 2.5}, LabelStyle → Black
]
```

Out[•]=



In[•]:=
```
StringFunc[s_] :=
 "κ=" <> ToString[NumberForm[Eκ[s] // N, {2, 2}]] <> ", κ′=" <> ToString[
   NumberForm[D[Eκ[t] // ComplexExpand, t] /. t → s // Evaluate // N, {2, 2}]] <>
  ", τ=" <> ToString[NumberForm[Eτ[s] // N, {2, 2}]] <> ", τ′=" <> ToString[
   NumberForm[D[Eτ[t] // ComplexExpand, t] /. t → s // Evaluate // N, {2, 2}]]
```

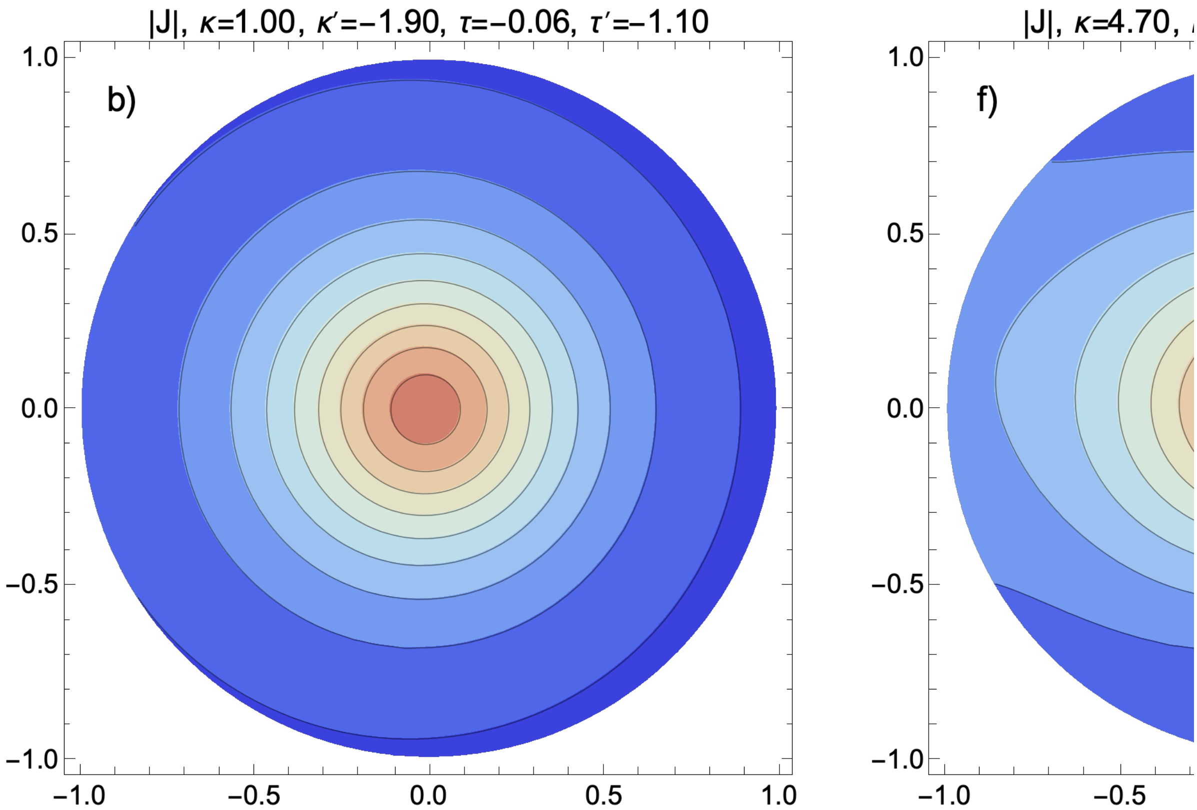
```
CSB = GraphicsRow[{
   ContourPlot[
    Norm[Bv[r, s, φ]] /. SetZeroAFunc //. ConfigE //. ConfigGH //. ConfigGeneral /.
         RCoords /. p1[t] → Sqrt[x^2 + y^2] /.
       p3[t] → ArcTan[x, y] /. p2[t] → 0 // Evaluate,
    {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
    Contours → Range[5, 16, 0.5], MaxRecursion → 2, ColorFunction →
     ColorData[{"ThermometerColors", {5, 16}}], ColorFunctionScaling → False,
    PlotLabel → "|B|, " <> StringFunc[0], ImageSize → {400, 400},
    ContourLabels → None, LabelStyle → {FontSize → 14, FontColor → Black},
    Epilog → {Text[Style["a)", 18], Scaled[{0.08, 0.92}]]},
    ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
   ],
   ContourPlot[
    Norm[Bv[r, s, φ]] /. SetZeroAFunc //. ConfigE //. ConfigGH //. ConfigGeneral /.
         RCoords /. p1[t] → Sqrt[x^2 + y^2] /.
       p3[t] → ArcTan[x, y] /. p2[t] → 0.75 // Evaluate,
    {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
    Contours → Range[5, 16, 0.5], MaxRecursion → 2, ColorFunction →
     ColorData[{"ThermometerColors", {5, 16}}], ColorFunctionScaling → False,
    PlotLabel → "|B|, " <> StringFunc[0.75], ImageSize → {400, 400},
    ContourLabels → None, LabelStyle → {FontSize → 14, FontColor → Black},
    Epilog → {Text[Style["e)", 18], Scaled[{0.08, 0.92}]]},
    ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
   ],
   ContourPlot[
    Norm[Bv[r, s, φ]] /. SetZeroAFunc //. ConfigE //. ConfigGH //. ConfigGeneral /.
         RCoords /. p1[t] → Sqrt[x^2 + y^2] /.
       p3[t] → ArcTan[x, y] /. p2[t] → 0.8 // Evaluate,
    {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
    Contours → Range[5, 16, 0.5], MaxRecursion → 2, ColorFunction →
     ColorData[{"ThermometerColors", {5, 16}}], ColorFunctionScaling → False,
    PlotLabel → "|B|, " <> StringFunc[0.8], ImageSize → {400, 400},
    ContourLabels → None, LabelStyle → {FontSize → 14, FontColor → Black},
    Epilog → {Text[Style["i)", 18], Scaled[{0.08, 0.92}]]},
    ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
   ],
   BarLegend[{"ThermometerColors", {5, 16}},
    Range[5, 16, 1], LegendMargins → 0, LegendLabel → "B [nT]",
    LabelStyle → {FontSize → 16, FontColor → Black}, LegendMarkerSize → 350]
  }, ImageSize → 1600, Alignment → Left
 ]
Export["/Users/ajefweiss/Desktop/cross_sections_b.png", CSB, "PNG"]
```

Out[ ]=


Out[ ]= /Users/ajefweiss/Desktop/cross_sections_b.png

```
CSJ = GraphicsRow[{
   ContourPlot[
    Norm[Jv[r, s, φ]] / 1.256 /. SetZeroAFunc //. ConfigE //. ConfigGH //.
          ConfigGeneral /. RCoords /. p1[t] → Sqrt[x^2 + y^2] /.
       p3[t] → ArcTan[x, y] /. p2[t] → 0 // Evaluate,
    {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
    Contours → Range[0, 500, 50], MaxRecursion → 2, ColorFunction →
     ColorData[{"ThermometerColors", {0, 500}}], ColorFunctionScaling → False,
    PlotLabel → "|J|, " <> StringFunc[0], ImageSize → {400, 400},
    ContourLabels → None, LabelStyle → {FontSize → 14, FontColor → Black},
    Epilog → {Text[Style["b)", 18], Scaled[{0.08, 0.92}]]},
    ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
   ],
   ContourPlot[
    Norm[Jv[r, s, φ]] / 1.256 /. SetZeroAFunc //. ConfigE //. ConfigGH //.
          ConfigGeneral /. RCoords /. p1[t] → Sqrt[x^2 + y^2] /.
       p3[t] → ArcTan[x, y] /. p2[t] → 0.75 // Evaluate,
    {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
    Contours → Range[0, 500, 50], MaxRecursion → 2, ColorFunction →
     ColorData[{"ThermometerColors", {0, 500}}], ColorFunctionScaling → False,
    PlotLabel → "|J|, " <> StringFunc[0.75], ImageSize → {400, 400},
    ContourLabels → None, LabelStyle → {FontSize → 14, FontColor → Black},
    Epilog → {Text[Style["f)", 18], Scaled[{0.08, 0.92}]]},
    ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
   ],
   ContourPlot[
    Norm[Jv[r, s, φ]] / 1.256 /. SetZeroAFunc //. ConfigE //. ConfigGH //.
          ConfigGeneral /. RCoords /. p1[t] → Sqrt[x^2 + y^2] /.
       p3[t] → ArcTan[x, y] /. p2[t] → 0.8 // Evaluate,
    {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
    Contours → Range[0, 500, 50], MaxRecursion → 2, ColorFunction →
     ColorData[{"ThermometerColors", {0, 500}}], ColorFunctionScaling → False,
    PlotLabel → "|J|, " <> StringFunc[0.8], ImageSize → {400, 400},
    ContourLabels → None, LabelStyle → {FontSize → 14, FontColor → Black},
    Epilog → {Text[Style["j)", 18], Scaled[{0.08, 0.92}]]},
    ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
   ],
   BarLegend[{"ThermometerColors", {0, 600}},
    Range[0, 600, 50], LegendMargins → 0, LegendLabel → "J [nA/m^2]",
    LabelStyle → {FontSize → 16, FontColor → Black}, LegendMarkerSize → 350]
  }, ImageSize → 1600, Alignment → Left
 ]
Export["/Users/ajefweiss/Desktop/cross_sections_j.png", CSJ, "PNG"]
```

Out[∘]=

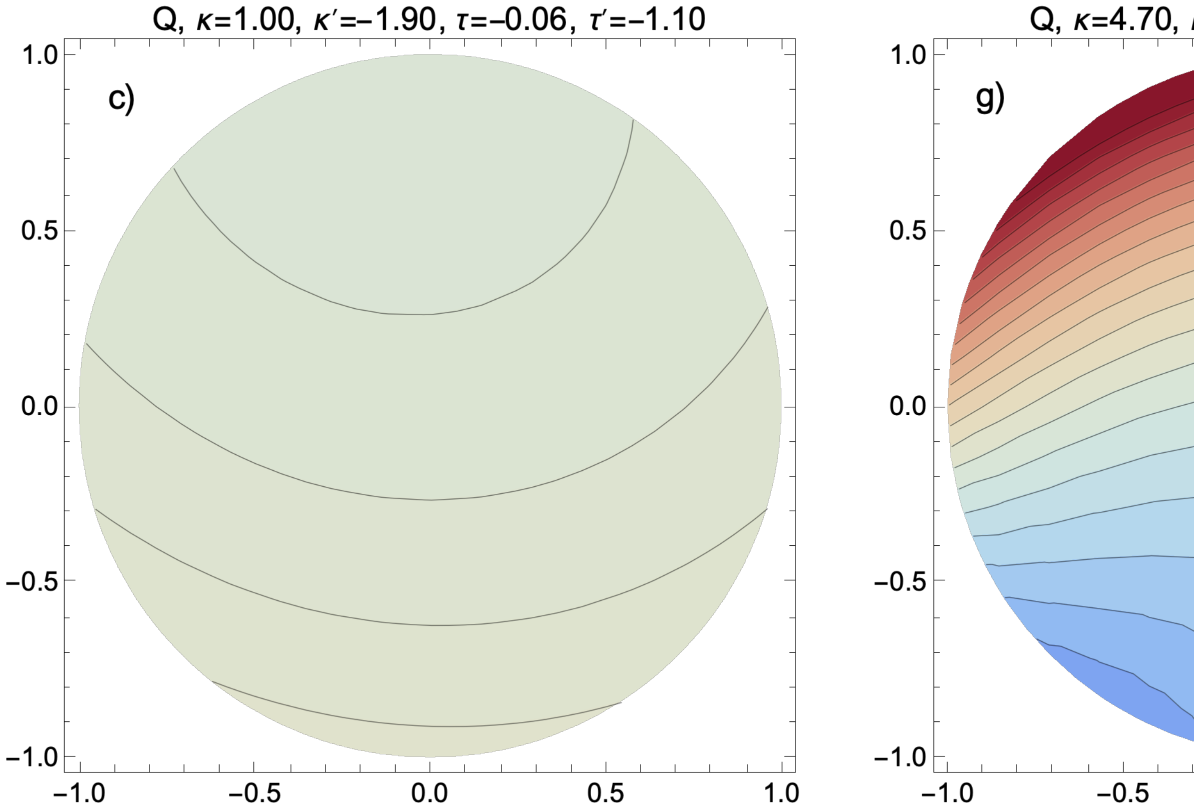


Out[∘]= /Users/ajefweiss/Desktop/cross_sections_j.png

In[∘]:=
```
BTwist =
  (r σ FSBφ[r, s, φ] + r σ τ[s] FSBs[r, s, φ]) / (r (1 + r σ κ[s] Cos[φ]) FSBs[r, s, φ]) /.
        l[s] → 1 /. SetZeroAFunc //. ConfigE //. ConfigGH //. ConfigGeneral /.
    RCoords /. p1[t] → Sqrt[x^2 + y^2] /. p3[t] → ArcTan[x, y];
```

```
In[•]:= CST = GraphicsRow[{
     ContourPlot[
      BTwist /. p2[t] → 0 // Evaluate,
      {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
      Contours → Range[1.4, 2.6, 0.01], MaxRecursion → 1,
      ColorFunction → ColorData[{"ThermometerColors", {1.4, 2.6}}],
      ColorFunctionScaling → False, PlotLabel → "Q, " <> StringFunc[0],
      ImageSize → {400, 400}, ContourLabels → None,
      LabelStyle → {FontSize → 14, FontColor → Black},
      Epilog → {Text[Style["c)", 18], Scaled[{0.08, 0.92}]]},
      ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
     ],
     ContourPlot[
      BTwist /. p2[t] → 0.75 // Evaluate,
      {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
      Contours → Range[1.4, 2.6, 0.05], MaxRecursion → 0,
      ColorFunction → ColorData[{"ThermometerColors", {1.4, 2.6}}],
      ColorFunctionScaling → False, PlotLabel → "Q, " <> StringFunc[0.75],
      ImageSize → {400, 400}, ContourLabels → None,
      LabelStyle → {FontSize → 14, FontColor → Black},
      Epilog → {Text[Style["g)", 18], Scaled[{0.08, 0.92}]]},
      ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
     ],
     ContourPlot[
      BTwist /. p2[t] → 0.8 // Evaluate,
      {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
      Contours → Range[1.4, 2.6, 0.05], MaxRecursion → 0,
      ColorFunction → ColorData[{"ThermometerColors", {1.4, 2.6}}],
      ColorFunctionScaling → False, PlotLabel → "Q, " <> StringFunc[0.8],
      ImageSize → {400, 400}, ContourLabels → None,
      LabelStyle → {FontSize → 14, FontColor → Black},
      Epilog → {Text[Style["k)", 18], Scaled[{0.08, 0.92}]]},
      ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
     ],
     BarLegend[{"ThermometerColors", {1.4, 2.6}},
      Range[1.4, 2.6, 0.05], LegendMargins → 0, LegendLabel → "Q",
      LabelStyle → {FontSize → 16, FontColor → Black}, LegendMarkerSize → 350]
    }, ImageSize → 1600, Alignment → Left
   ]
   Export["/Users/ajefweiss/Desktop/cross_sections_twist.png", CST, "PNG"]
```

Out[∘]=

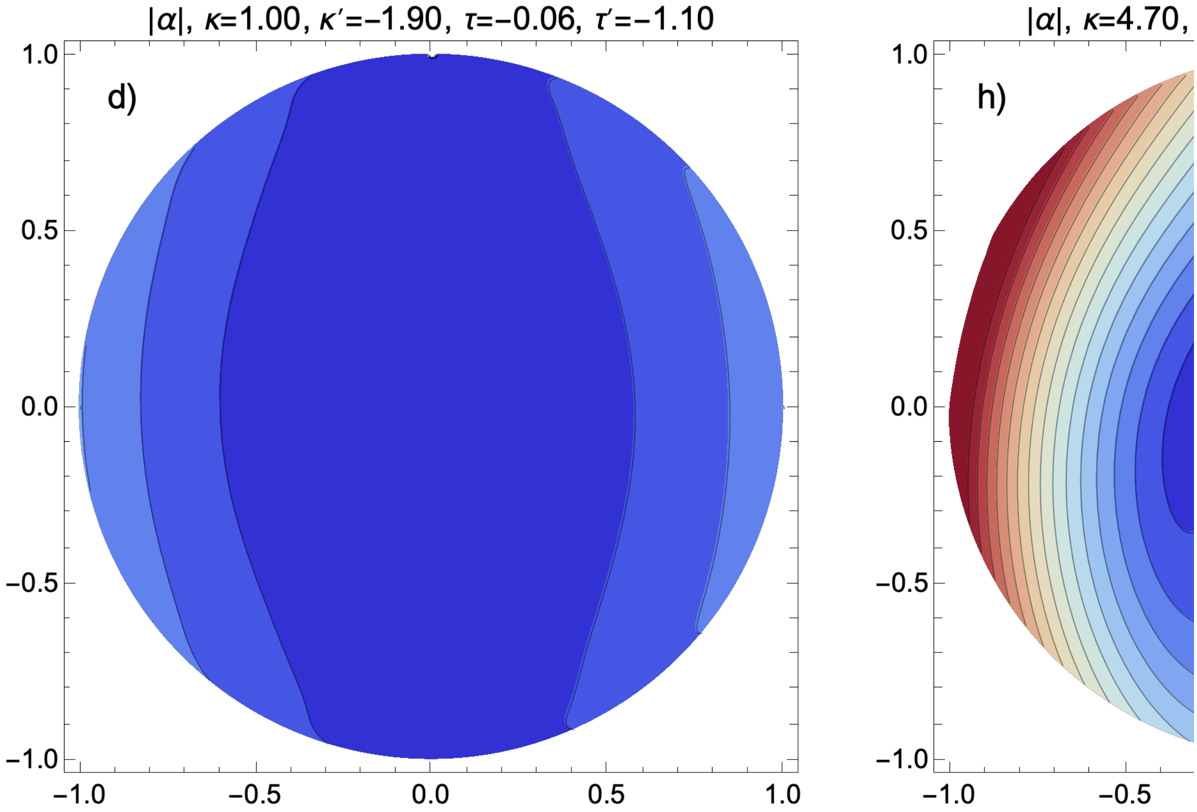


Out[∘]= /Users/ajefweiss/Desktop/cross_sections_twist.png

In[∘]:=
```
BJAlign =
  180 / Pi ArcSin[Norm[Bv[r, s, φ] × Jv[r, s, φ]] / Norm[Bv[r, s, φ]] / Norm[Jv[r, s,
            φ]]] /. SetZeroAFunc //. ConfigE //. ConfigGH //. ConfigGeneral /.
     RCoords /. p1[t] → Sqrt[x^2 + y^2] /. p3[t] → ArcTan[x, y];
```

```
In[*]:= CSJxB = GraphicsRow[{
    ContourPlot[
     BJAlign /. p2[t] → 0 // Evaluate,
     {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
     Contours → Range[0, 30, 2], MaxRecursion → 2, ColorFunction →
      ColorData[{"ThermometerColors", {0, 30}}], ColorFunctionScaling → False,
     PlotLabel → "|α|, " <> StringFunc[0.], ImageSize → {400, 400},
     ContourLabels → None, LabelStyle → {FontSize → 14, FontColor → Black},
     Epilog → {Text[Style["d)", 18], Scaled[{0.08, 0.92}]]},
     ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
    ],
    ContourPlot[
     BJAlign /. p2[t] → 0.75 // Evaluate,
     {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
     Contours → Range[0, 30, 2], MaxRecursion → 2, ColorFunction →
      ColorData[{"ThermometerColors", {0, 30}}], ColorFunctionScaling → False,
     PlotLabel → "|α|, " <> StringFunc[0.75], ImageSize → {400, 400},
     ContourLabels → None, LabelStyle → {FontSize → 14, FontColor → Black},
     Epilog → {Text[Style["h)", 18], Scaled[{0.08, 0.92}]]},
     ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
    ],
    ContourPlot[
     BJAlign /. p2[t] → 0.8 // Evaluate,
     {x, -1, 1}, {y, -1, +1}, RegionFunction → Function[{x, y}, x^2 + y^2 < 1],
     Contours → Range[0, 30, 2], MaxRecursion → 2, ColorFunction →
      ColorData[{"ThermometerColors", {0, 30}}], ColorFunctionScaling → False,
     PlotLabel → "|α|, " <> StringFunc[0.8], ImageSize → {400, 400},
     ContourLabels → None, LabelStyle → {FontSize → 14, FontColor → Black},
     Epilog → {Text[Style["l)", 18], Scaled[{0.08, 0.92}]]},
     ImageSize → Large, LabelStyle → {FontSize → 24, FontColor → Black}
    ],
    BarLegend[{"ThermometerColors", {0, 30}},
     Range[0, 30, 2], LegendMargins → 0, LegendLabel → "α [deg]",
     LabelStyle → {FontSize → 16, FontColor → Black}, LegendMarkerSize → 350]
   }, ImageSize → 1600, Alignment → Left
  ]
Export["/Users/ajefweiss/Desktop/cross_sections_jxb.png", CSJxB, "PNG"]
```

Out[•]=



Out[•]= /Users/ajefweiss/Desktop/cross_sections_jxb.png

In[•]:=
```
FRP = (FSsqg[r, s, φ] FSϵr[r, s, φ] FSFr[r, s, φ] /. SetZeroAFunc // Expand) /.
    IntegralTermsNullRulesResult /. IntegralTermsReplacementRulesResult;
FSP = (FSsqg[r, s, φ] FSϵs[r, s, φ] FSFs[r, s, φ] /. SetZeroAFunc // Expand) /.
    IntegralTermsNullRulesResult /. IntegralTermsReplacementRulesResult;
FPP = (FSsqg[r, s, φ] FSϵφ[r, s, φ] FSFφ[r, s, φ] /. SetZeroAFunc // Expand) /.
    IntegralTermsNullRulesResult /. IntegralTermsReplacementRulesResult;

FRPN = Coefficient[FRP //. ConfigGH //. ConfigGeneral, n[s]];
FRPB = Coefficient[FRP //. ConfigGH //. ConfigGeneral, b[s]];

FSPT = Coefficient[FSP //. ConfigGH //. ConfigGeneral, t[s]];
FSPN = Coefficient[FSP //. ConfigGH //. ConfigGeneral, b[s]];
FSPB = Coefficient[FSP //. ConfigGH //. ConfigGeneral, n[s]];

FPPN = Coefficient[FPP //. ConfigGH //. ConfigGeneral, n[s]];
FPPB = Coefficient[FPP //. ConfigGH //. ConfigGeneral, b[s]];
```

ReplaceAll: {IntegralTermsNullRulesResult} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

ReplaceAll: {IntegralTermsReplacementRulesResult} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

ReplaceAll: {IntegralTermsNullRulesResult} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

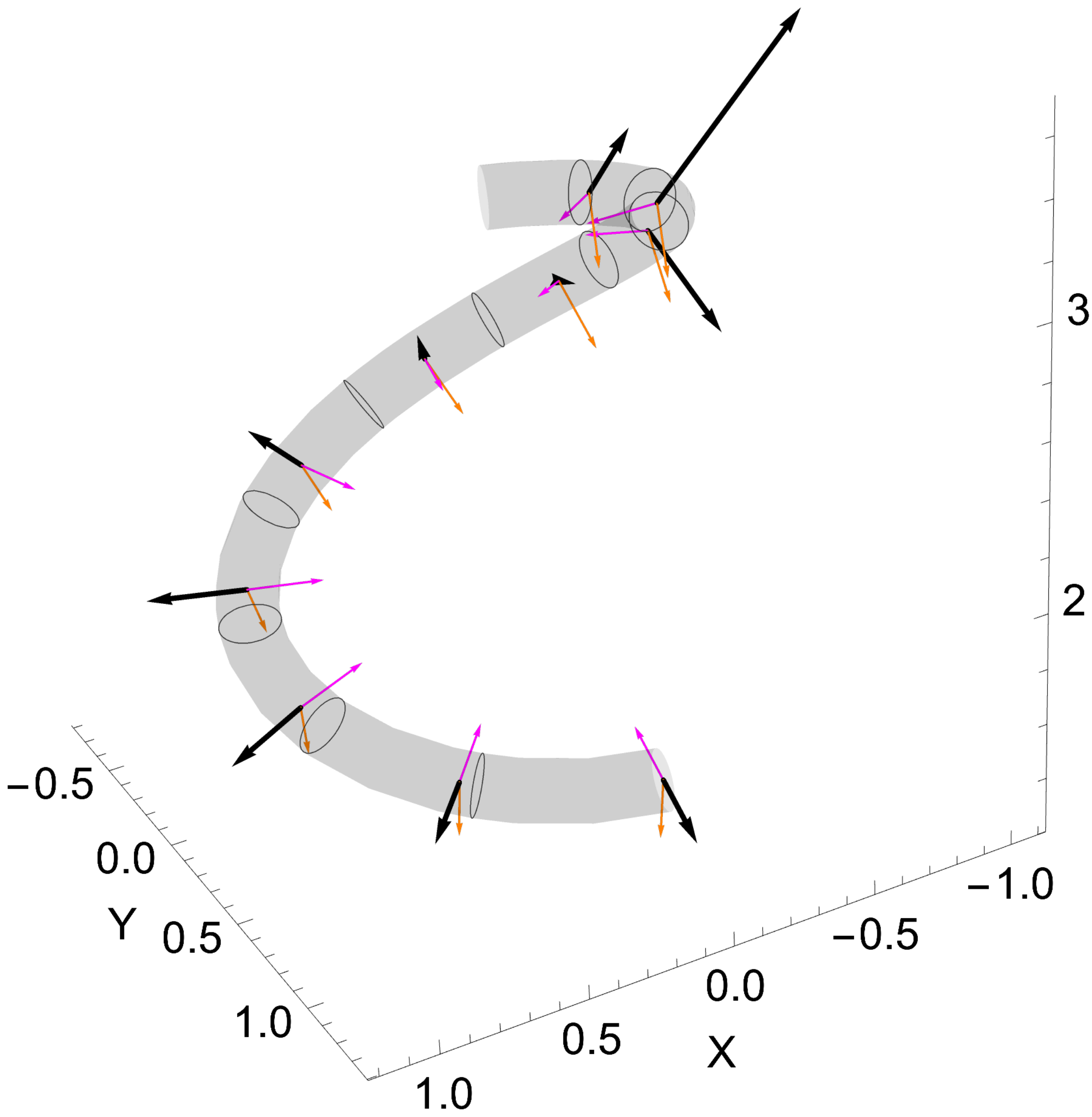
General: Further output of ReplaceAll::reps will be suppressed during this calculation.

```
In[•]:= sVs = Range[0, 1, .1];
FRNVs = {};
FRBVs = {};

FSTVs = {};
FSNVs = {};
FSBVs = {};

FPNVs = {};
FPBVs = {};

For[i = 1, i < Length[sVs], i++,
 {
  Print["Computing for s=", sVs[[i]]];
  rnv = NIntegrate[
    FRPN /. ConfigE /. s → sVs[[i]] // Evaluate, {r, 0, 1}, AccuracyGoal → 10];
  rbv = NIntegrate[
    FRPB /. ConfigE /. s → sVs[[i]] // Evaluate, {r, 0, 1}, AccuracyGoal → 10];

  stv = NIntegrate[
    FSPT /. ConfigE /. s → sVs[[i]] // Evaluate, {r, 0, 1}, AccuracyGoal → 10];
  snv = NIntegrate[
    FSPN /. ConfigE /. s → sVs[[i]] // Evaluate, {r, 0, 1}, AccuracyGoal → 10];
  sbv = NIntegrate[
    FSPB /. ConfigE /. s → sVs[[i]] // Evaluate, {r, 0, 1}, AccuracyGoal → 10];

  pnv = NIntegrate[
    FPPN /. ConfigE /. s → sVs[[i]] // Evaluate, {r, 0, 1}, AccuracyGoal → 10];
  pbv = NIntegrate[
    FPPB /. ConfigE /. s → sVs[[i]] // Evaluate, {r, 0, 1}, AccuracyGoal → 10];

  FRNVs = AppendTo[FRNVs, rnv];
  FRBVs = AppendTo[FRBVs, rbv];

  FSTVs = AppendTo[FSTVs, stv];
  FSNVs = AppendTo[FSNVs, snv];
  FSBVs = AppendTo[FSBVs, snv];

  FPNVs = AppendTo[FPNVs, pnv];
  FPBVs = AppendTo[FPBVs, pbv];
 }
]
```

```
Computing for s=0.

Computing for s=0.1

Computing for s=0.2

Computing for s=0.3

Computing for s=0.4

Computing for s=0.5

Computing for s=0.6

Computing for s=0.7

Computing for s=0.8

Computing for s=0.9
```

In[∘]:=
```
FArrows = {};

For[i = 1, i < Length[sVs], i++, {
  Fvec = ((FSTVs⟦i⟧) Et[s] + (FRNVs⟦i⟧ + FSNVs⟦i⟧ + FPNVs⟦i⟧) En[s] +
      (FRBVs⟦i⟧ + FSBVs⟦i⟧ + FPBVs⟦i⟧) Eb[s]) / 15;
  FArrows = AppendTo[FArrows, Graphics3D[{Arrowheads[.02], Thickness[.005],
      Black, {Arrow[{Eγ[s], Eγ[s] + Fvec} /. s → sVs⟦i⟧ /. ConfigGeneral]}}]
   ];
  FArrows =
   AppendTo[FArrows, Graphics3D[{Arrowheads[.01], Thickness[.002], Magenta,
      {Arrow[{Eγ[s], Eγ[s] + 2.5 σ En[s]} /. s → sVs⟦i⟧ /. ConfigGeneral]}}]
   ];
  FArrows =
   AppendTo[FArrows, Graphics3D[{Arrowheads[.01], Thickness[.002], Orange,
      {Arrow[{Eγ[s], Eγ[s] + 2.5 σ Eb[s]} /. s → sVs⟦i⟧ /. ConfigGeneral]}}]
   ];
 }]

Show[
 {
  ParametricPlot3D[Er[s] /. ConfigGeneral /. r → 1 // Evaluate, {s, 0, 1},
   {φ, 0, 2 Pi}, PlotStyle → {Opacity[0.1], Black}, Mesh → {10, 0}],
  FArrows
 },
 AxesLabel → {"X", "Y", "Z"}, Boxed → False, TicksStyle → Large,
 AxesStyle → Large, ViewPoint → {2, 4, 2.5}, LabelStyle → Black, PlotRange → All
]
```

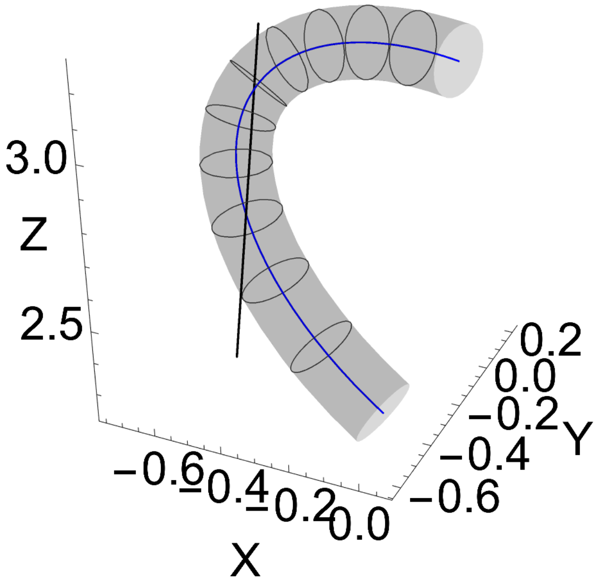
Out[ ]=



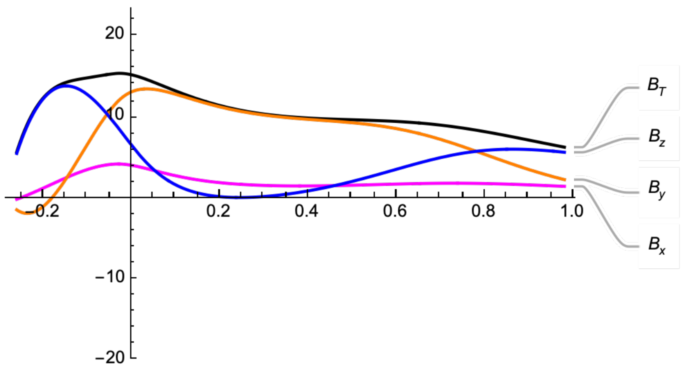
In[ ]:=
```
pfunc[repr_] := Er[s] /. ConfigGeneral /. repr
bfunc[repr_] :=
  Bv[r, s, φ] /. SetZeroAFunc //. ConfigE //. ConfigGH //. ConfigGeneral /. repr
```

```
In[•]:= p1 = Er[s] /. ConfigGeneral /. r → 0.2 /. s → 0.75 /. φ → Pi;
p2 = Er[s] /. ConfigGeneral /. r → 1 /. s → 0.6 /. φ → Pi / 4;

p[t_] := p1 + (p2 - p1) t

coords = {};
tparams = {};
trange = Range[-0.5, 1.5, 0.02];
For[i = 1, i < Length[trange], i++,
 {
  sol = NMinimize[{Norm[(Er[s] - p[trange⟦i⟧] /. ConfigGeneral)] // Evaluate,
     r ∈ PositiveReals, s ∈ PositiveReals, φ ∈ PositiveReals} , {r, s, φ}];
  If[(r < 1 /. sol⟦2⟧) && (sol⟦1⟧ < 0.001),
   coords = AppendTo[coords, sol⟦2⟧]; tparams = AppendTo[tparams, trange⟦i⟧]];
 }
]

Show[
 {
  ParametricPlot3D[Eϒ[s] /. ConfigGeneral // Evaluate,
   {s, .5, 1}, PlotStyle → {Opacity[1], Blue, Thickness[.004]}],
  ParametricPlot3D[Er[s] /. ConfigGeneral /. r → 1 // Evaluate, {s, .5, 1},
   {φ, 0, 2 Pi }, PlotStyle → {Opacity[0.15], Black}, Mesh → {10, 0}],
  ParametricPlot3D[p[t] // Evaluate, {t, -0.5, 1.5},
   PlotStyle → {Opacity[1], Black, Thickness[.005]}]
 },
 AxesLabel → {"X", "Y", "Z"}, Boxed → False, TicksStyle → Large, AxesStyle → Large,
 ViewPoint → {2, -4, 2.5}, LabelStyle → Black, PlotRange → All
]
fx = Interpolation[
   {tparams, Thread[bfunc[coords]]⟦1⟧} // Transpose, InterpolationOrder → 3];
fy = Interpolation[
   {tparams, Thread[bfunc[coords]]⟦2⟧} // Transpose, InterpolationOrder → 3];
fz = Interpolation[
   {tparams, Thread[bfunc[coords]]⟦3⟧} // Transpose, InterpolationOrder → 3];

Plot[{Norm[{fx[t], fy[t], fz[t]}], fx[t], fy[t], fz[t]},
 {t, Min[tparams], Max[tparams]}, PlotRange → {-20, 20},
 PlotStyle → {Black, Magenta, Orange, Blue}, PlotLabels → {"B_T", "B_x", "B_y", "B_z"}]
```

Out[ ]=

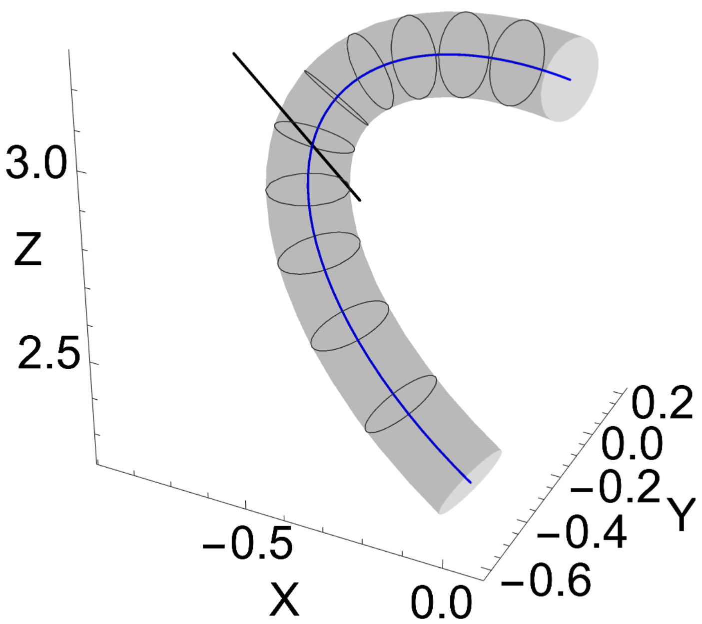

3.0
Z
2.5
−0.6
−0.4
−0.2
0.0
X
0.2
0.0
−0.2
−0.4
−0.6
Y

Out[ ]=

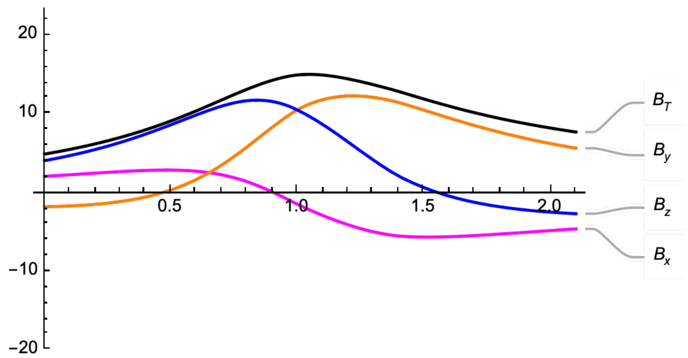

20
10
−10
−20
−0.2
0.2
0.4
0.6
0.8
1.0
B_T
B_z
B_y
B_x

In[•]:=
```
p2b = Er[s] /. ConfigGeneral /. r → 0 /. s → 0.72 /. φ → 0;
p1b = Er[s] /. ConfigGeneral /. r → 1 /. s → 0.74 /. φ → Pi / 5;

pb[t_] := p1b + (p2b - p1b) t

coordsb = {};
tparamsb = {};
trangeb = Range[-1.5, 2.5, 0.1];
For[i = 1, i < Length[trangeb], i++,
 {
  solb = NMinimize[{Norm[(Er[s] - pb[trangeb⟦i⟧] /. ConfigGeneral)] // Evaluate,
     r ∈ PositiveReals, s ∈ PositiveReals, φ ∈ PositiveReals} , {r, s, φ}];
  If[(r < 1 /. solb⟦2⟧) && (solb⟦1⟧ < 0.001), coordsb = AppendTo[coordsb, solb⟦2⟧];
   tparamsb = AppendTo[tparamsb, trangeb⟦i⟧]];
 }
]
Show[
 {
  ParametricPlot3D[Eϒ[s] /. ConfigGeneral // Evaluate,
   {s, .5, 1}, PlotStyle → {Opacity[1], Blue, Thickness[.004]}],
  ParametricPlot3D[Er[s] /. ConfigGeneral /. r → 1 // Evaluate, {s, .5, 1},
   {φ, 0, 2 Pi }, PlotStyle → {Opacity[0.15], Black}, Mesh → {10, 0}],
  ParametricPlot3D[pb[t] // Evaluate, {t, -1.5, 2.5},
   PlotStyle → {Opacity[1], Black, Thickness[.005]}]
 },
 AxesLabel → {"X", "Y", "Z"}, Boxed → False, TicksStyle → Large, AxesStyle → Large,
 ViewPoint → {2, -4, 2.5}, LabelStyle → Black, PlotRange → All
]
fxb = Interpolation[
   {tparamsb, Thread[bfunc[coordsb]]⟦1⟧} // Transpose, InterpolationOrder → 3];
fyb = Interpolation[
   {tparamsb, Thread[bfunc[coordsb]]⟦2⟧} // Transpose, InterpolationOrder → 3];
fzb = Interpolation[
   {tparamsb, Thread[bfunc[coordsb]]⟦3⟧} // Transpose, InterpolationOrder → 3];

Plot[{Norm[{fxb[t], fyb[t], fzb[t]}], fxb[t], fyb[t], fzb[t]},
 {t, Min[tparamsb], Max[tparamsb]}, PlotRange → {-20, 20},
 PlotStyle → {Black, Magenta, Orange, Blue}, PlotLabels → {"B_T", "B_x", "B_y", "B_z"}]
```

Out[∘]=

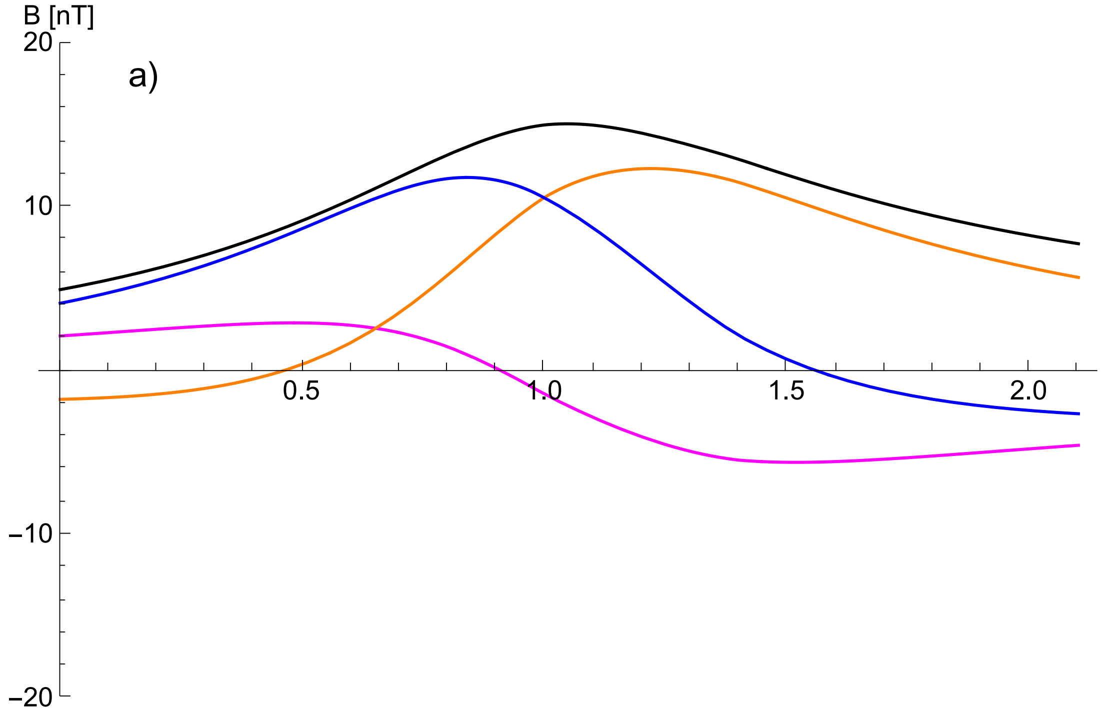


Out[∘]=



In[∘]:=
```
GraphicsRow[
 {
  Plot[
   {Norm[{fxb[t + Min[tparamsb]], fyb[t + Min[tparamsb]], fzb[t + Min[tparamsb]]}],
    fxb[t + Min[tparamsb]], fyb[t + Min[tparamsb]], fzb[t + Min[tparamsb]]},
   {t, 0, Max[tparamsb] - Min[tparamsb]}, PlotRange → {- 20, 20},
   PlotStyle → {Black, Magenta, Orange, Blue},
   AxesLabel → {"", "B  [nT]"}, PlotLegends → {"BT", "BX", "BY", "BZ"},
   Epilog → {Text[Style["a)", 18], Scaled[{0.1, 0.95}]]},
   ImageSize → Large, LabelStyle → {FontSize → 14, FontColor → Black}],
  Plot[{Norm[{fx[t + Min[tparams]], fy[t + Min[tparams]], fz[t + Min[tparams]]}],
    fx[t + Min[tparams]], fy[t + Min[tparams]], fz[t + Min[tparams]]},
   {t, 0, Max[tparams] - Min[tparams]}, PlotRange → {-20, 20},
   PlotStyle → {Black, Magenta, Orange, Blue}, AxesLabel → {"", "B  [nT]"},
   Epilog → {Text[Style["b)", 18], Scaled[{0.1, 0.95}]]},
   ImageSize → Large, LabelStyle → {FontSize → 14, FontColor → Black}]

 }, ImageSize → 1400
]
```

Out[∘]=



In[∘]:=
```
Show[
 {
  ParametricPlot3D[Eγ[s] /. ConfigGeneral // Evaluate,
   {s, .5, 1}, PlotStyle → {Opacity[1], Blue, Thickness[.004]}],
  ParametricPlot3D[Er[s] /. ConfigGeneral /. r → 1 // Evaluate, {s, .5, 1},
   {φ, 0, 2 Pi }, PlotStyle → {Opacity[0.15], Black}, Mesh → {10, 0}],
  ParametricPlot3D[p[t] // Evaluate, {t, -1.5, 2.5},
   PlotStyle → {Opacity[1], Black, Thickness[.005]}],
  ParametricPlot3D[pb[t] // Evaluate, {t, -2, 4},
   PlotStyle → {Opacity[1], Black, Thickness[.005]}],
  Graphics3D[Text[Style["b)", 18], p[-1] + {.12, 0, 0}]],
  Graphics3D[Text[Style["a)", 18], pb[-2.5] + {-.1, 0, 0}]]
 },
 AxesLabel → {"X", "Y", "Z"}, Boxed → False, TicksStyle → Large, AxesStyle → Large,
 ViewPoint → {2, -4, 2.5}, LabelStyle → Black, PlotRange → All
]
```

Out[∘]=