Chemical gradient walk of Lablets in chemical field at steady state

Abhishek Sharma
Ruhr Universität, Bochum
Germany

Please cite : Autonomous lablet locomotion and active docking by sensomotory electroosmotic drive
Abhishek Sharma and John S. McCaskill,
Proceedings of the European Conference on Artificial Life 2015, pp. 456-462

Description

In this *Mathematica* notebook, we consider dynamics of self propelled particle (Lablet) in the chemical field. This formulation was taken from two references,

1) Johannes Taktikos, Vasily Zaburdaev, and Holger Stark, Modeling a self-propelled autochemotactic walker, Phys. Rev. E 84, 041924

2) Johannes Taktikos, Vasily Zaburdaev, and Holger Stark, Collective dynamics of model microorganisms with chemotactic signaling, Phys. Rev. E 85, 051901

Chemotaxis with constant speed in constant field

We assume Lablet moves with constant speed, hence in two dimensions, velocity at any general time is given by $V(t) = v\, e(t)$.

We assume chemotatic field is given by, E. So, we could define the driving potential as
$V(e) = -e.E$.

Based on this, we can write governing equation of the motion of the particle for rotational and kinematic motion as,

$$\frac{d}{dt} L = -\gamma_R \Omega + M_{\text{ext}} + \mathrm{T}(t)$$

$$\frac{d}{dt} e = \Omega \times e$$

Using these equations, in the overdamped limit (neglecting inertial effects), we can write

$$\frac{d}{dt} e = \frac{1}{\gamma_R} (\mathcal{I} - e \otimes e) E + \frac{1}{\gamma_R} \mathrm{T}(t) \times e$$

Substituting in this equation, $e = (\cos\phi,\ \sin\phi,\ 0)^T$, $E = (E_x, E_y, 0)^T$, $\mathrm{T}(t) = (0, 0, \mathrm{T}(t))^T$, the equation simplifies to,

$$\frac{d}{dt} \phi(t) = -\frac{E_x}{\gamma_R} \sin\phi(t) + \frac{E_y}{\gamma_R} \cos\phi(t) + \sqrt{2 q_\phi}\ \mathrm{T}(t)$$

Constant chemotactic field along X direction

In the absense of noise and constant chemotact field along x direction,

$$\frac{d}{dt} \phi(t) = -\frac{E_x}{\gamma_R} \sin\phi(t)$$

In[1]:=
```
eq = D[ϕ[t], t] == - Ex/γR Sin[ϕ[t]];
```

In[2]:=
```
ic = ϕ[0] == ϕ0;
```

In[3]:=
```
sol = DSolve[{eq, ic}, ϕ[t], t]
```

Solve::ifun :
Inverse functions are being used by Solve, so some solutions may not be found; use Reduce for complete solution information. >>

Solve::ifun :
Inverse functions are being used by Solve, so some solutions may not be found; use Reduce for complete solution information. >>

Out[3]= $\left\{\left\{\phi[t] \to 2\,\mathrm{ArcCot}\left[e^{\frac{\mathrm{Ex}\,t}{\gamma R}} \mathrm{Cot}\left[\frac{\phi 0}{2}\right]\right]\right\}\right\}$

Hence, the trajectory of the particle is given by,

$$\frac{d}{dt} r(t) = v\, e(t)$$

In[4]:=
```
solX = DSolve[{D[x[t], t] == v Cos[ϕ[t]] /. First[sol], x[0] == 0}, x[t], t]
```

Out[4]= $\left\{\left\{x[t] \to -\frac{1}{\mathrm{Ex}} v \left(\mathrm{Ex}\, t + \gamma R\, \mathrm{Log}[2] - \gamma R\, \mathrm{Log}\left[1 + e^{\frac{2\,\mathrm{Ex}\,t}{\gamma R}} - \mathrm{Cos}[\phi 0] + e^{\frac{2\,\mathrm{Ex}\,t}{\gamma R}} \mathrm{Cos}[\phi 0]\right]\right)\right\}\right\}$

In[5]:=
```
solY = DSolve[{D[y[t], t] == v Sin[ϕ[t]] /. First[sol], y[0] == 0}, y[t], t]
```

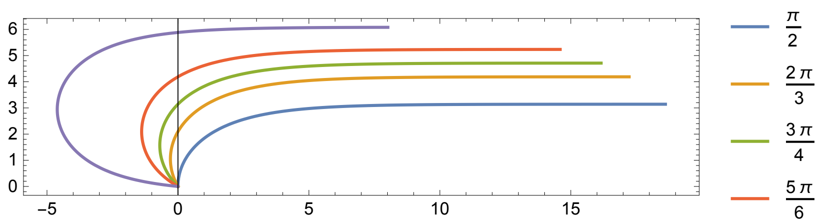
Out[5]= $\left\{\left\{y[t] \to -\frac{1}{\mathrm{Ex}} 2 \left(v\, \gamma R\, \mathrm{ArcTan}\left[\mathrm{Cot}\left[\frac{\phi 0}{2}\right]\right] - v\, \gamma R\, \mathrm{ArcTan}\left[e^{\frac{\mathrm{Ex}\,t}{\gamma R}} \mathrm{Cot}\left[\frac{\phi 0}{2}\right]\right]\right)\right\}\right\}$

In[6]:=
```
params = {Ex → 1., γR → 2., v → 1.0};
```

In[7]:=
```
f[ϕ0_] = Evaluate[{x[t] /. First[solX], y[t] /. First[solY]} /. params]
```

Out[7]=
$$\left\{-1.\left(1.38629+1.\,t-2.\,\mathrm{Log}\left[1+e^{1.\,t}-\mathrm{Cos}[\phi 0]+e^{1.\,t}\mathrm{Cos}[\phi 0]\right]\right),\ -2.\left(2.\,\mathrm{ArcTan}\left[\mathrm{Cot}\left[\frac{\phi 0}{2}\right]\right]-2.\,\mathrm{ArcTan}\left[e^{0.5\,t}\,\mathrm{Cot}\left[\frac{\phi 0}{2}\right]\right]\right)\right\}$$

In[8]:=
$$\mathtt{ParametricPlot}\left[\mathtt{Evaluate}\left[\mathtt{f[\#]\ \&\ /@}\left\{\frac{\pi}{2},\frac{2\pi}{3},\frac{3\pi}{4},\frac{5\pi}{6},\pi-0.1\right\}\right],\ \{t,0,20\},\ \mathtt{Frame}\to\mathtt{True},\ \mathtt{PlotLegends}\to\left\{\frac{\pi}{2},\frac{2\pi}{3},\frac{3\pi}{4},\frac{5\pi}{6}\right\}\right]$$

Out[8]=


Case 2: Two dimensional field from coplanar electrodes (using conformal mapping)

Conformal map to coplanar geometry from parallel plate geometry

In[9]:=
```
Clear[t, t1, t2, t3, t4]
```

In[10]:=
```
(* Changing from the T Space to U Space *)
```
$$\mathtt{f0[t\_]} = \int \left(C \Big/ \left(\sqrt{\left((t-t1)\,(t-t2)\,(t-t3)\,(t-t4)\right)}\right)\right) dt;$$

In[11]:=
```
FullSimplify[f0[t]]
(* Simplifying this gives the following Form,
the constant C is calculated from the complete Integral, F(t3) = 1 *)
```

Out[11]=
$$\left(2\,C\,(t-t2)\sqrt{\frac{(-t1+t2)\,(t-t3)}{(t-t1)\,(t2-t3)}}\,(t-t4)\ \mathrm{EllipticF}\left[\mathrm{ArcSin}\left[\sqrt{\frac{(t-t2)\,(t1-t4)}{(t-t1)\,(t2-t4)}}\right],\frac{(t1-t3)\,(t2-t4)}{(t2-t3)\,(t1-t4)}\right]\right) \Big/ \left(\sqrt{(t-t1)\,(t-t2)\,(t-t3)\,(t-t4)}\sqrt{\frac{(t-t2)\,(-t1+t2)\,(t-t4)\,(t1-t4)}{(t-t1)^2\,(t2-t4)^2}}\,(t2-t4)\right)$$

In[12]:=
$$f[t\_] = \frac{2\,C}{\sqrt{(t2-t3)\,(t1-t4)}}\,\text{EllipticF}\left[\text{ArcSin}\left[\sqrt{\frac{(t-t2)\,(t1-t4)}{(t-t1)\,(t2-t4)}}\,\right],\ \frac{(t1-t3)\,(t2-t4)}{(t2-t3)\,(t1-t4)}\right];$$

$$f1[t\_] = \text{EllipticF}\left[\text{ArcSin}\left[\sqrt{\frac{(t-t2)\,(t1-t4)}{(t-t1)\,(t2-t4)}}\,\right],\ \frac{(t1-t3)\,(t2-t4)}{(t2-t3)\,(t1-t4)}\right];$$

In[14]:=
$$aa = \text{Chop}\left[\text{EllipticF}\left[\text{ArcSin}\left[\sqrt{\frac{(t3-t2)\,(t1-t4)}{(t3-t1)\,(t2-t4)}}\,\right],\ \frac{(t1-t3)\,(t2-t4)}{(t2-t3)\,(t1-t4)}\right]\right];$$

In[15]:=
$$g[t\_] := \text{ArcSin}\left[\sqrt{\frac{(t-t2)\,(t1-t4)}{(t-t1)\,(t2-t4)}}\,\right];$$

$$p = \frac{(t1-t3)\,(t2-t4)}{(t2-t3)\,(t1-t4)};$$

```
(* Complete Elliptical Integral*)
iK = EllipticK[p];
t1 = -2; t2 = -0.4; t3 = 0.4; t4 = 2; (* Assumed *)
```

$$FF[t\_] = \frac{f1[t]}{aa};$$

In[20]:=
```
p1 = ParametricPlot[With[{ω = u + I v}, {Re[ω], Im[ω]}],
    {u, 0, 1}, {v, 0, 1}, Mesh → Full, MeshStyle → {Red}];
```

In[21]:=
```
p2 = ParametricPlot[With[{ω = u + I v, b = FF[u + I v] }, {Re[b], Im[b]}], {u, -1, 1},
    {v, 0, 1}, Mesh → Full, MeshStyle → {Red}, PlotRange → {0., 0.9}];
```

In[22]:=
```
(* Trying to make the Invert Plots for the same *)
```

In[23]:=
```
FK[u_, v_] = Flatten[Solve[f1[t] == aa (u + I v), t]][[1, 2]];
```

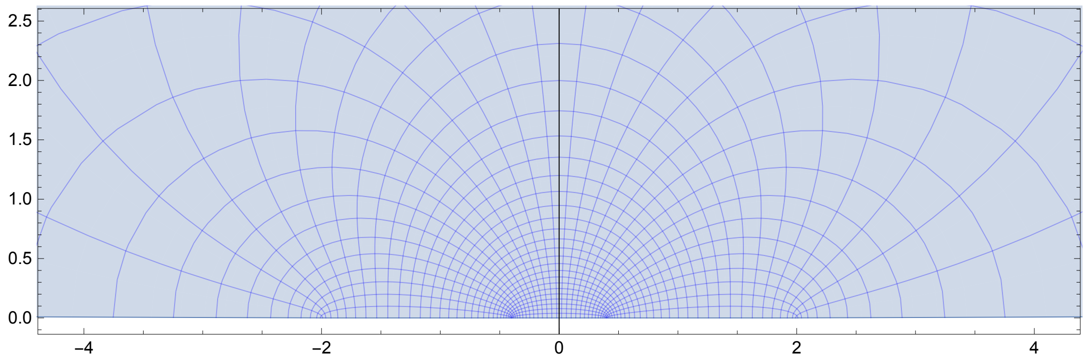
Solve::ifun :

Inverse functions are being used by Solve, so some solutions may not be found; use Reduce for complete solution information. ≫

In[24]:=
```
p3 = ParametricPlot[With[{ω = u + I v, b = FK[u, v] }, {Re[b], Im[b]}], {u, 0, 1},
    {v, 0, 0.95}, MeshStyle → {Red}, Mesh → {20, 20}, AspectRatio → 0.6];
```

In[25]:=
```
ParametricPlot[With[{ω = u + I v, b = FK[u, v]}, {Re[b], Im[b]}],
 {u, 0, 1}, {v, 0, 0.95}, Mesh → {30, 30}, MeshStyle → Blue]
```

Out[25]=



Here, we will use one dimensional conformally invariant steady state solution for fast chemical reaction at the electrode. Conformal invariance could be found in Bazant et al, 2004.

In[26]:=
```
V = 10.;
J = Tan[V/4.];
Conc[ω_] := 1 + J Re[ω];
K = 0.1;
Pot[ω_] := Log[(1 + J Re[ω])/(K (1 - J^2))];
```

In[31]:=
```
intp1 =
 Interpolation[Flatten[Table[{{Re[Sin[FK[u, v]]], Im[Sin[FK[u, v]]]}, Conc[u + I v]},
    {u, 0, 1, 0.01}, {v, 0, 0.8, 0.01}], 1]]
```

Interpolation::udeg : Interpolation on unstructured grids is currently only supported for InterpolationOrder->1 or InterpolationOrder->All. Order will be reduced to 1. >>

Out[31]= InterpolatingFunction[Domain: {{−18.8, 18.8}, {−3.46, 28.1}} Output: scalar]

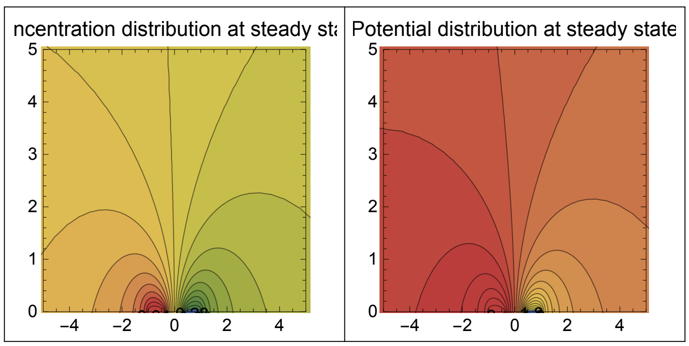
Solution for coplanar geometry

In[32]:=
```
p1 = ListContourPlot[
   Flatten[Table[{Re[Sin[FK[u, v]]], Im[Sin[FK[u, v]]], Conc[u + I v] },
     {u, 0, 1, 0.01}, {v, 0, 0.8, 0.01}], 1],
   Contours → 20, ContourLabels → All, ColorFunction → "DarkRainbow",
   PlotLabel → "Concentration distribution at steady state",
   PlotRange → {{-5, 5}, {0, 5}, All}];
```

In[33]:=
```
p2 = ListContourPlot[
    Flatten[Table[{Re[Sin[FK[u, v]]], Im[Sin[FK[u, v]]], Pot[u + I v] },
      {u, 0, 1, 0.01}, {v, 0, 0.8, 0.01}], 1],
    Contours → 20, ContourLabels → All, ColorFunction → "DarkRainbow",
    PlotLabel → "Potential distribution at steady state",
    PlotRange → {{-5, 5}, {0, 5}, All}];
```

In[34]:=
```
GraphicsRow[{p1, p2}, Frame → All]
```

Out[34]=



In[35]:=
```
intp1 =
  Interpolation[Flatten[Table[{{Re[Sin[FK[u, v]]], Im[Sin[FK[u, v]]]}, Conc[u + I v]},
     {u, 0, 1, 0.01}, {v, 0, 0.8, 0.01}], 1]]
```

Interpolation::udeg : Interpolation on unstructured grids is currently only supported for InterpolationOrder−>1 or InterpolationOrder−>All. Order will be reduced to 1. >>

Out[35]= InterpolatingFunction[Domain: {{−18.8, 18.8}, {−3.46, 28.1}} Output: scalar]

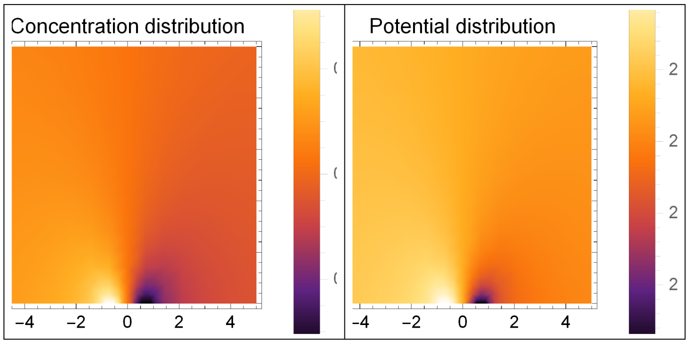
In[36]:=
```
intp2 =
  Interpolation[Flatten[Table[{{Re[Sin[FK[u, v]]], Im[Sin[FK[u, v]]]}, Pot[u + I v]},
     {u, 0, 1, 0.01}, {v, 0, 0.8, 0.01}], 1]]
```

Interpolation::udeg : Interpolation on unstructured grids is currently only supported for InterpolationOrder−>1 or InterpolationOrder−>All. Order will be reduced to 1. >>

Out[36]= InterpolatingFunction[Domain: {{−18.8, 18.8}, {−3.46, 28.1}} Output: scalar]

In[37]:=
```
GraphicsRow[{DensityPlot[intp1[i, j], {i, -5, 5}, {j, 0, 5}, PlotRange → All,
    PlotLabel → "Concentration distribution", ColorFunction → "SunsetColors",
    PlotLegends → Automatic], DensityPlot[intp2[i, j], {i, -5, 5},
    {j, 0, 5}, PlotRange → All, PlotLabel → "Potential distribution",
    ColorFunction → "SunsetColors", PlotLegends → Automatic]}, Frame → All]
```

Out[37]=



In[38]:=
```
dInpt1X[X_?NumericQ, Y_?NumericQ] = D[intp1[X, Y], X]
```

Out[38]= InterpolatingFunction[Domain: {{-18.8, 18.8}, {-3.46, 28.1}} Output: scalar][X, Y]

In[39]:=
```
dInpt1Y[X_?NumericQ, Y_?NumericQ] = D[intp1[X, Y], Y]
```

Out[39]= InterpolatingFunction[Domain: {{-18.8, 18.8}, {-3.46, 28.1}} Output: scalar][X, Y]

In[40]:=
```
dInpt1X2[X_?NumericQ, Y_?NumericQ] = D[D[intp1[X, Y], X], X]
```

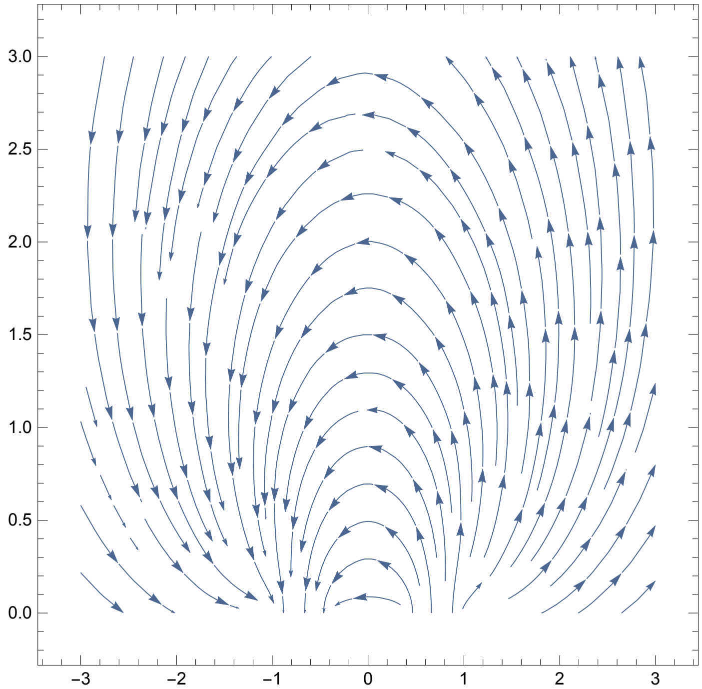
Out[40]= InterpolatingFunction[Domain: {{-18.8, 18.8}, {-3.46, 28.1}} Output: scalar][X, Y]

In[41]:=
```
dInpt1Y2[X_?NumericQ, Y_?NumericQ] = D[D[intp1[X, Y], Y], Y]
```

Out[41]= InterpolatingFunction[Domain: {{-18.8, 18.8}, {-3.46, 28.1}} Output: scalar][X, Y]

In[42]:= `StreamPlot[{dInpt1X[X, Y], dInpt1Y[X, Y]}, {X, -3, 3}, {Y, 0, 3}]`

Out[42]=



In[43]:= `Clear[eq, eq1, eq2]`

In[44]:= `eq1 =`

$$D[\phi[t], t] == -\frac{dInpt1X[pX[t], pY[t]]}{\gamma R} Sin[\phi[t]] + \frac{dInpt1Y[pX[t], pY[t]]}{\gamma R} Cos[\phi[t]];$$

In[45]:= `eq2 = D[pX[t], t] == v Cos[ϕ[t]];`

In[46]:= `eq3 = D[pY[t], t] == v Sin[ϕ[t]];`

In[47]:= `ic1 = ϕ[0] == ϕ0;`

In[48]:= `ic2 = pX[0] == pX0;`

In[49]:= `ic3 = pY[0] == pY0;`

In[50]:= `params = {γR → 0.2, v → 0.005, pX0 → 2.0, pY0 → 2.0, ϕ0 → π/3};`

In[51]:=
```
solN = NDSolve[Evaluate[{eq1, eq2, eq3, ic1, ic2, ic3} /. params],
   {pX[t], pY[t], ϕ[t]}, {t, 0, 1550}]
```

Out[51]=
```
{{pX[t] → InterpolatingFunction[ Domain: {{0., 1.55×10^3}} Output: scalar ][t],
  pY[t] → InterpolatingFunction[ Domain: {{0., 1.55×10^3}} Output: scalar ][t],
  ϕ[t] → InterpolatingFunction[ Domain: {{0., 1.55×10^3}} Output: scalar ][t]}}
```

In[52]:=
```
{pX[0], pY[0]} /. solN
```

Out[52]=
```
{{pX[0], pY[0]}}
```

In[53]:=
```
px1 = ParametricPlot[{pX[t], pY[t]} /. solN, {t, 0, 1550},
    Frame → True, PlotRange → {{-5, 5}, {-5, 10}}, PlotStyle → Dotted];
```

In[54]:=
```
Show[DensityPlot[intp1[i, j], {i, -5, 5}, {j, 0, 5}, PlotRange → All,
   ColorFunction → "SunsetColors", PlotLegends → Automatic],
  ParametricPlot[{pX[t], pY[t]} /. solN, {t, 0, 1550}, Frame → True,
   PlotRange → {{-5, 5}, {-5, 10}}, PlotStyle → {Blue, Dashed, Thick}],
  StreamPlot[{dInpt1X[X, Y], dInpt1Y[X, Y]}, {X, -5, 5},
   {Y, 0, 5}, StreamPoints → 50, StreamStyle → Black],
  ListPlot[({pX[t], pY[t]} /. solN) /. t → 0, PlotStyle → {Green, PointSize[0.03]}],
  ListPlot[({pX[t], pY[t]} /. solN) /. t → 1550, PlotStyle → {Red, PointSize[0.03]}]]
```

Out[54]=