

The Quantum Orbits Dynamic Dashboard

Introduction

The Quantum Orbits Dynamic Dashboard is a *Mathematica* application to aid in the visualization and understanding of the role of complex variables - specifically, complex time and complex position - in the semiclassical theory of tunnel ionization.

References and mild legalities

The Dashboard was developed as an aid for the research presented in

- 1
- Slalom in complex time: emergence of low-energy structures in tunnel ionization via complex time contours. E. Pisanty and M. Ivanov. In preparation.

and it is also documented in

- 2
- Quantum Orbit Dynamic Dashboard: a navigation tool for complex time and complex space in tunnel ionization. E. Pisanty. In preparation.

If this software is useful for your research, please cite either or both papers, or cite this software directly. An example citation is

- 3
- E. Pisanty. QuODD: Quantum Orbits Dynamic Dashboard. <https://github.com/episanty/QuODD> (2015).

This software is available under the MIT license, with the exception of the file Quantum Orbits Dynamic Dashboard.cdf, which is licensed under the Creative Commons Attribution-ShareAlike (CC BY-SA) license.

Table of contents

This document is structured as follows:

- **Motivation**, describing the quantities
- **The Dashboard** itself.
- **Dashboard Elements**, with a detailed explanation of each component.
- **Interaction with the Dashboard**, detailing the different controls.
- **How to make new Dashboards**.
- **Some physics examples**.

Export CDF

Export PDF

Motivation

Tunnel ionization occurs when an atom or molecule of ionization potential $I_p = \frac{1}{2} \kappa^2$ is ionized by a laser of low frequency ω and high peak field F . In the regime where the adiabaticity parameter $\gamma = \omega \kappa / F$ is small, the field is best thought of as oscillating slowly, thus providing a potential energy barrier which the atomic electrons can then tunnel through. Somewhat surprisingly, the tunnelling process can still be thought of in terms of trajectories, but this comes at the expense of having a negative kinetic energy and therefore, necessarily, a complex-valued velocity.

In the standard theory, the ionization happens at a complex time t_s which satisfies the energy conservation condition

$$\frac{1}{2} (\boldsymbol{p} + \boldsymbol{A}(t_s))^2 + I_p = 0.$$

Here \boldsymbol{p} is the electron's final momentum and $\boldsymbol{A}(t)$ is the laser field's vector potential, which we take to be linearly polarized and monochromatic (i.e. $\boldsymbol{A}(t) = -\frac{E}{\omega} \boldsymbol{z} \sin(\omega t)$); their sum $\boldsymbol{p} + \boldsymbol{A}(t)$ is the electron's velocity at time t . The electron trajectory is simply the integral of this velocity, from the ionization time t_s to any other time t , either real or complex:

$$\boldsymbol{r}_{\text{cl}}(t) = \int_{t_s}^t (\boldsymbol{p} + \boldsymbol{A}(\tau)) \, \mathrm{d}\tau$$

The Quantum Orbits Dynamic Dashboard explores precisely this trajectory, with a particular eye to the effect that different ways to cross the time plane have on the complex position.

One particular quantity of interest that can be derived from the complex trajectory is its Coulomb interaction with the ion it leaves behind. This Coulomb interaction has a potential energy

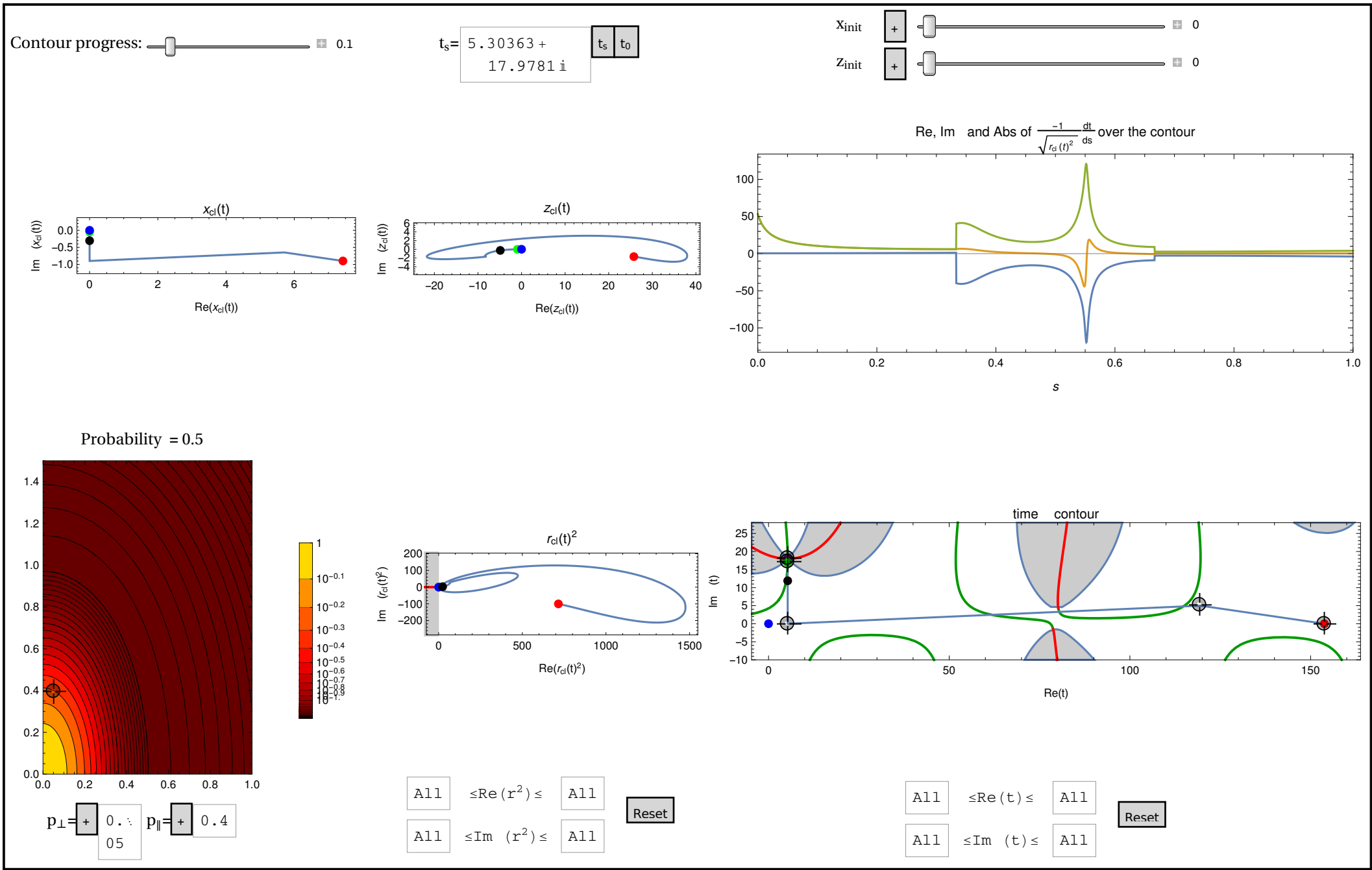
$$U(\boldsymbol{r}_{\text{cl}}(t)) = \frac{-1}{\sqrt{r_{\text{cl}}(t)^2}},$$

and it can be used to refine estimates of the tunnelling probability, which involve in particular the action induced by this energy, $\int_{t_s}^T U(\boldsymbol{r}_{\text{cl}}(t)) \, \mathrm{d}t$, over some complex contour that joins the ionization time t_s and the final detection time T . However, when doing this integral one must take care with the square root, which has a branch cut - a sign discontinuity - when its argument is real and negative. Integration contours on the time plane cannot cross such branch cuts, as they affect the value of the resulting integral; the Dashboard is an integral help in designing contours which avoid the cuts.

The Dashboard

Without further ado, the Dashboard looks like this:


dashboardPlotter[{{0.05, 0.055}, 1.007, {"tx", "t0", "t0" + $\frac{2 \pi}{0.055} + 5 \, \text{i}$, "t0" + $1.3 \frac{2 \pi}{0.055}$ }, {0.05, 0.4}]

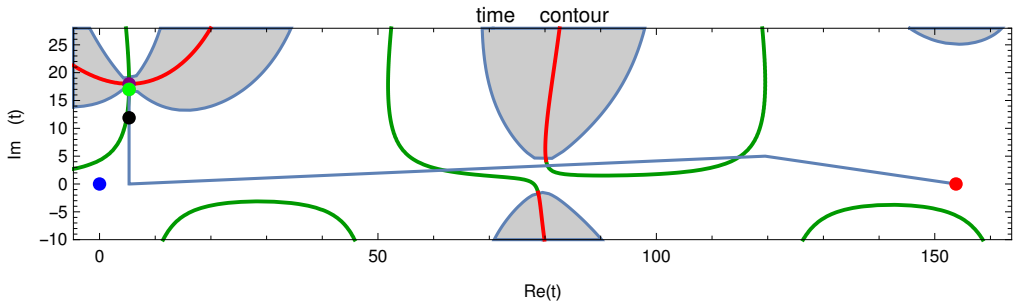


2 |

Dashboard elements

The time plane

The heart of the Dashboard is the time plane at lower right. It displays the complex time plane and, most importantly, the integration contour under consideration, in blue. Most importantly, the contour can be modified using the selectors marked 



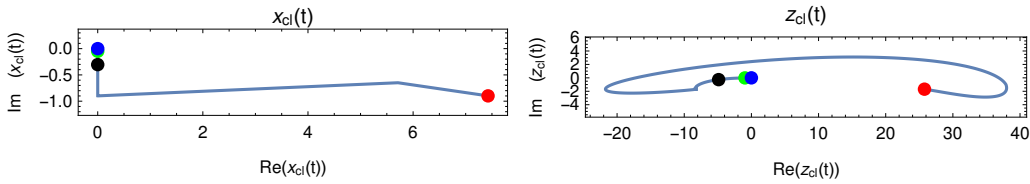
In addition, the time plane shows other relevant structures which describe the complex-valued function $r_{cl}(t)^2$:

- The branch cuts of the square root, i.e. the times for which $r_{cl}(t)^2$ is real and negative, are shown as red lines.
- By contrast, the green lines show the times for which $r_{cl}(t)^2$ is real and positive, which is in some ways the most desirable condition.
- The gray regions show the places where the real part of the squared position is negative: $\text{Re}(r_{cl}(t)^2) < 0$. This signifies that a branch cut is nearby, and can cause problems for ionic potentials more complex than the Coulomb interaction.

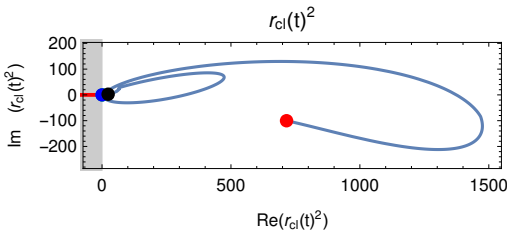
In this plot, and throughout, tooltips mark every relevant structure. Coloured dots indicate the start and end of the contour (green and red), the time origin $t=0$ (blue), the ionization time t_s (purple), as well as a manipulatable black dot on the contour controlled at the top left of the Dashboard.

Trajectory plots

The effect of the contour is seen most clearly on the semiclassical trajectory, $r_{cl}(t)$, and this is displayed component-wise on the top left and centre. Two components are important: the x component, transverse polarization, which is a scaled copy of the time contour (since $x_{cl}(t) = p_x(t - t_s)$), and the z component along the laser field, which displays most of the interesting dynamics.



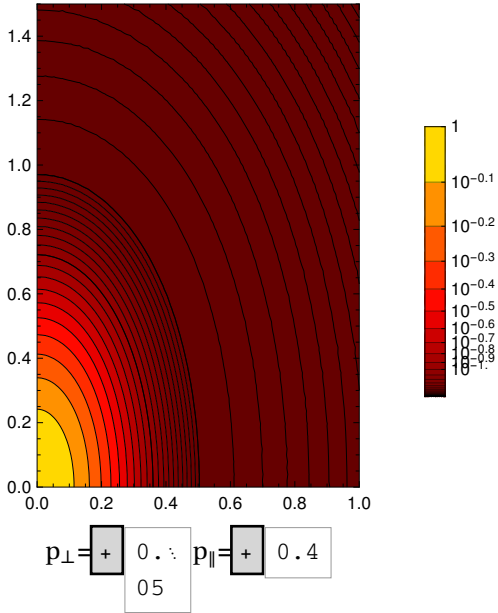
Position-squared plot



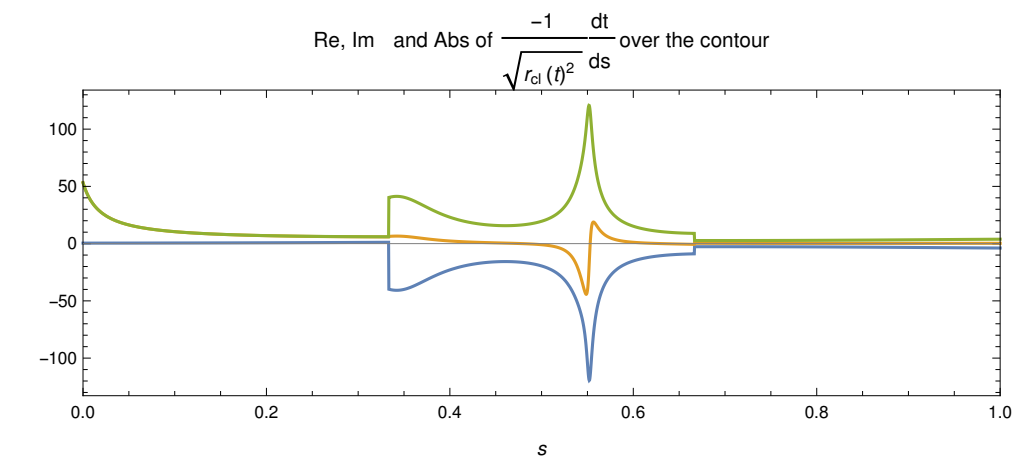
Momentum plane

$$e^{-2\text{Im}\left(\int_b^t \left[l_p + \frac{1}{2}(\mathbf{p} + \mathbf{A}(\tau))^2\right] d\tau\right)}$$

Probability = 0.5




Coulomb Integrand



The multiple elements of the Dashboard are:

`Graphics[Locator[Dynamic[{0, 0}, Null &, UpdateInterval -> Infinity]], ImageSize -> {17, 17}]`










- Three graphs which show the complex trajectory $\boldsymbol{r}_{\text{cl}}(t) = \int_{t_{\text{s}}}^t (\boldsymbol{p} + \boldsymbol{A}(\tau)) \, d\tau$ in the complex x , z and r^2 planes (top left, top centre and bottom centre, respectively).
- On bottom right, the time contour in the complex time plane is the blue line.
- On bottom left, the relative probability of ionization as a function of momentum, with the current value of momentum as a selector (). This shows exclusively the tunnelling probability, $e^{-2 \, \text{Im} \left(\int_{t_{\text{s}}}^t I_p + \frac{1}{2} (\boldsymbol{p} + \boldsymbol{A}(\tau))^2 \, d\tau \right)}$, normalized to 1 at the maximum of the distribution, which is at $\boldsymbol{p} = 0$.
- On top right, the integrand for the Coulomb correction as a function of normalized path length, in its real, imaginary and absolute parts.

Some more details and extra information:

- There's tooltips all over the place, so hover your mouse over stuff to try and find out what it is. Try, for example,
 - dots in the trajectories and the time contour
 - gray regions
 - the green and red lines on top and bottom right
 - the curves on bottom centre
 - the contours on bottom left show the value of probability
- The gray regions on bottom centre and right represent regions where r^2 and $r_{\text{cl}}(t)^2$, resp., have a negative real part. This is not catastrophic but these regions should be avoided; for example, molecular correlation potentials must not be trusted there.
- The red lines on top and bottom right are the lines where r^2 and $r_{\text{cl}}(t)^2$, resp., become real and negative; here $\sqrt{r^2}$ has a branch cut.
- The green lines, on the other hand, are the positive real axis of r^2 and $r_{\text{cl}}(t)^2$, where everything is fine.
- Dots on the top and on bottom right indicate the origin of each plot, the start and end of the contour, and a controllable point along it. The start of the contour is usually $t_k = t_{\text{s}} - i / \kappa^2$, which can be identified as the tunnel entrance.
- The colour scale on bottom left is linear but the contours are logarithmic. Thus there is a drop of one order of magnitude between successive thick lines and a drop of $\sqrt[10]{10}$ between successive thin lines (which also mark colour changes).
- This contour plot is normalized to the probability at the peak of the field, to zero total momentum.
- The contour on bottom right is parametrized uniformly between the different dots. This causes the rate of change $\frac{dt}{ds}$ to change from one segment to the next, which is what causes the abrupt shifts in the graph on top right.
- The integral with respect to s of the real and imaginary parts on bottom centre gives the invariant $\int_{\text{contour}} \frac{1}{\sqrt{r_{\text{cl}}(t)^2}} \, dt$, which is the desired Coulomb correction up to constants.

How to control the Dashboard

- Most importantly, the time contour itself is editable, by dragging the selectors marked  on the kinks of the contour, or by clicking near them (the nearest selector moves).
- The momentum can also be changed by dragging the selector marked  on the contour plot on bottom left, or by clicking anywhere on that plot.
- For finer control, the text fields below the momentum plot can be used to input specific values for each component. This can also be used to enter values outside those shown on the plot.
- The sign of both components can be changed by clicking the  button, which will turn it to a  and then back.
- The slider marked Contour progress on the top moves a black dot along the contour the time plane, and along the corresponding space plots.
- The r^2 and time plots, on bottom centre and bottom right, have adjustable plot ranges below them. Click  to set them to the default setting.
- The ionization time t_{s} , at which the trajectory $\boldsymbol{r}_{\text{cl}}(t) = \int_{t_{\text{s}}}^t (\boldsymbol{p} + \boldsymbol{A}(\tau)) \, d\tau$ starts, can be controlled using the input box at top centre. Clicking the  button sets it to the saddle point time (which obeys $\frac{1}{2} (\boldsymbol{p} + \boldsymbol{A}(t_{\text{s}}))^2 + I_p = 0$), and the  button sets it to the real part of that. This is useful for investigating the process of deforming the ionization-time integration contour up to the saddle-point time.
- The sliders and controls on top right add and control an initial position to the classical position, $\boldsymbol{r}_{\text{cl}}(t) = \boldsymbol{r}_{\text{init}} + \int_{t_{\text{s}}}^t (\boldsymbol{p} + \boldsymbol{A}(\tau)) \, d\tau$. This is required if the start time is set to zero, as the trajectory is meant to start on the ARM boundary.

How to call the function

To make a new Dashboard, you can use the `dashboardPlotter` function. This requires full-blown *Mathematica* and cannot be done on the CDF player. To see the calling syntax, use

`?dashboardPlotter`

<code>dynamicDashboardPlotter</code> <code>[[F, ω], κ]</code> plots a dashboard for field amplitude <code>F</code> at frequency ω , for ionization potential $\kappa^2/2$.
<code>dynamicDashboardPlotter</code> <code>[[F, ω], κ, path]</code> institutes the desired path, where the strings "t κ ", "t s ", "t0" and " τ " will be replaced by the corresponding functions of momentum , and "T" is a laser period . Default is {"t κ ", "t0", "T"}.
<code>dynamicDashboardPlotter</code> <code>[[F, ω], κ, path, {pointit , ppinit }]</code> specifies initial values of pointit and pp init for p_{\perp} and p_{\parallel} .

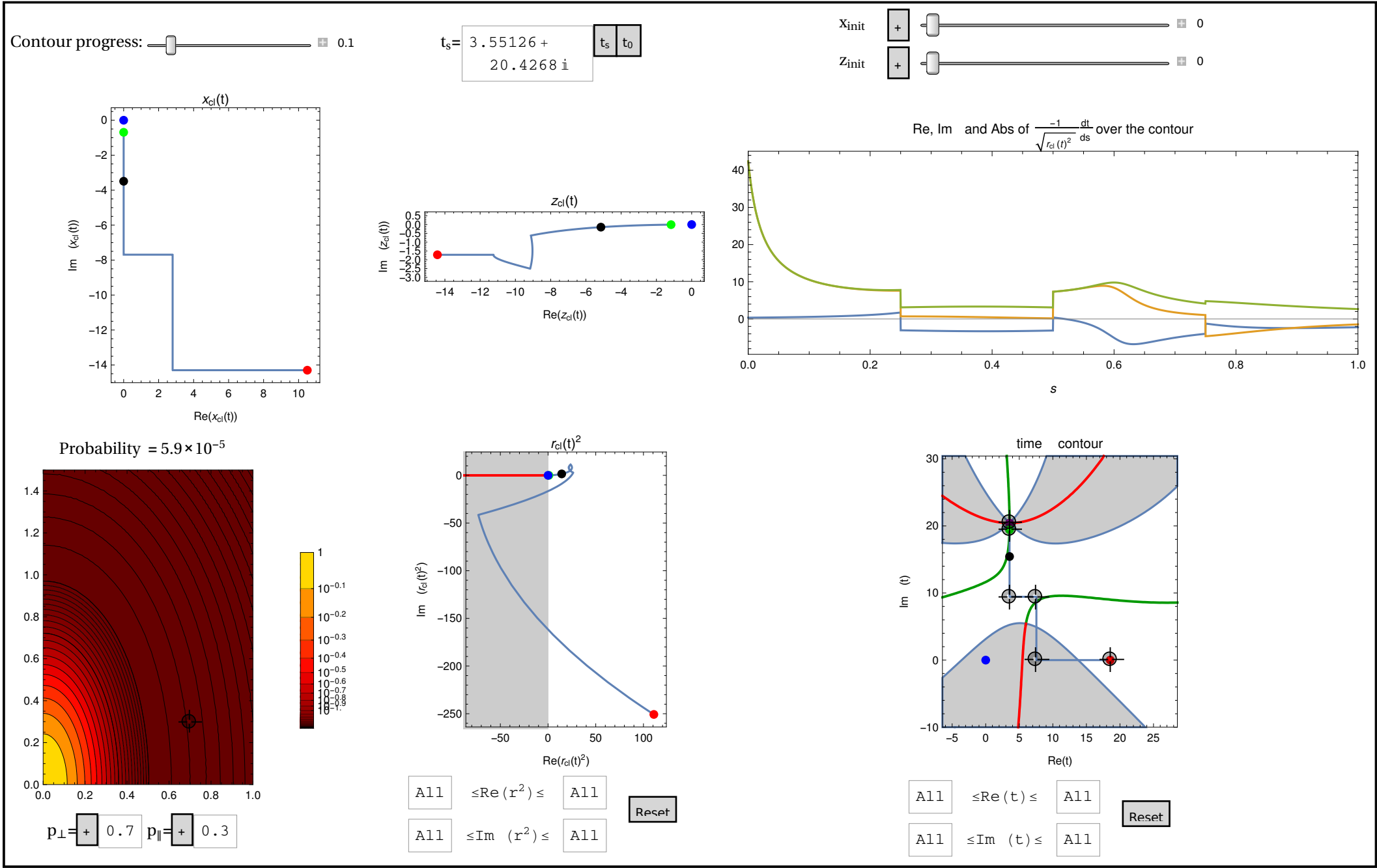
Entering +k will pull up templates for convenience (possibly not in V8, though, or it may need ++k or something).

Some physics examples

The ‘standard’ contour can fail

because branch cuts of $\sqrt{r_{\text{cl}}(t)^2}$ can cross the real axis. Unfortunately, this specific case occurs out in the wings of the distribution where nothing gets observed to begin with. (In essence, the transverse momentum for this case is implausibly large. This causes the transverse coordinate to become very imaginary, and the longitudinal coordinate cannot offset the large and negative $x_{\text{cl}}(t)^2$, so the sum becomes negative.)

4 | dashboardPlotter[{0.05, 0.055}, 1.007, {"tκ", "tκ"-10 i, "tκ"-10 i+4, "t0"+4, "t0"+15}, {0.7, 0.3}]



There are two important drawbacks with this scenario, that I can see:

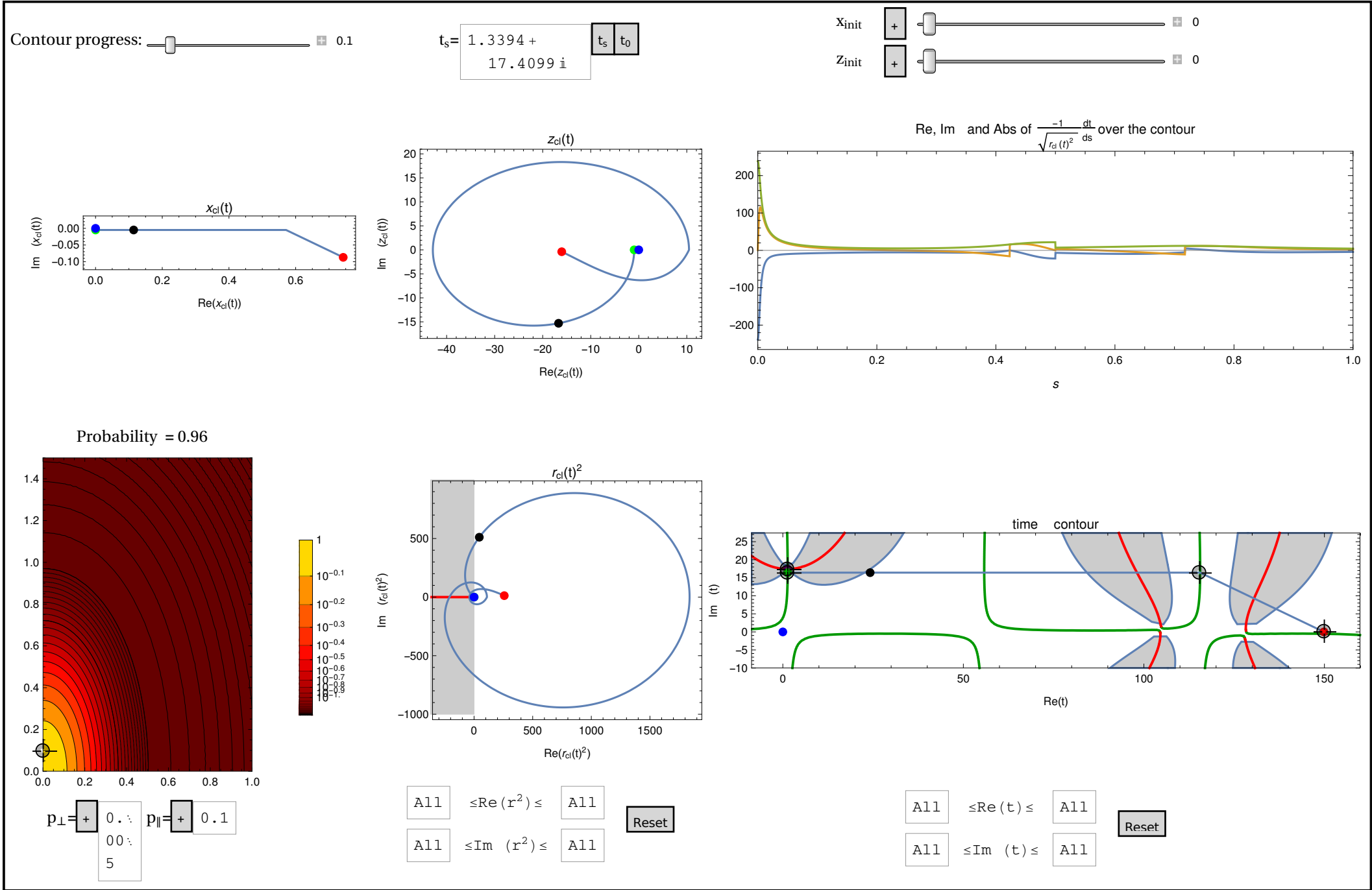
- It only happens at very low probability. The starting point for this is for transverse momentum just above $p_{\perp} = 0.5$ for 'standard' conditions, and zero longitudinal momentum. This means that the ionization probability there is low (less than 1%), and who knows whether other effects come in to cloud this picture or not. Higher p_{\perp} make the behaviour a lot clearer but this really drives the amplitude down.
- There is still a horizontal bit of the contour before the drop, and this still accumulates phase as any energy the electron has is being integrated along a real time direction. This may or may not mean that the time delay is not actually measurable, but it merits careful consideration.

Recolliding electrons

... look quite interesting with these tools.

Consider a 'standard' recolliding electron (zero transverse momentum and small, reasonable and positive momentum along the laser polarization) and take the time contour through slightly more than one period. During the recollision, it crosses the branch cut twice!

dashboardPlotter[{0.05, 0.055}, 1.007, {"tκ", "tκ"+1. $\frac{2\pi}{0.055}$, "t0"+1.3 $\frac{2\pi}{0.055}$ }, {0.005, 0.1}]



There are two aspects of this behaviour which look very robust to me.

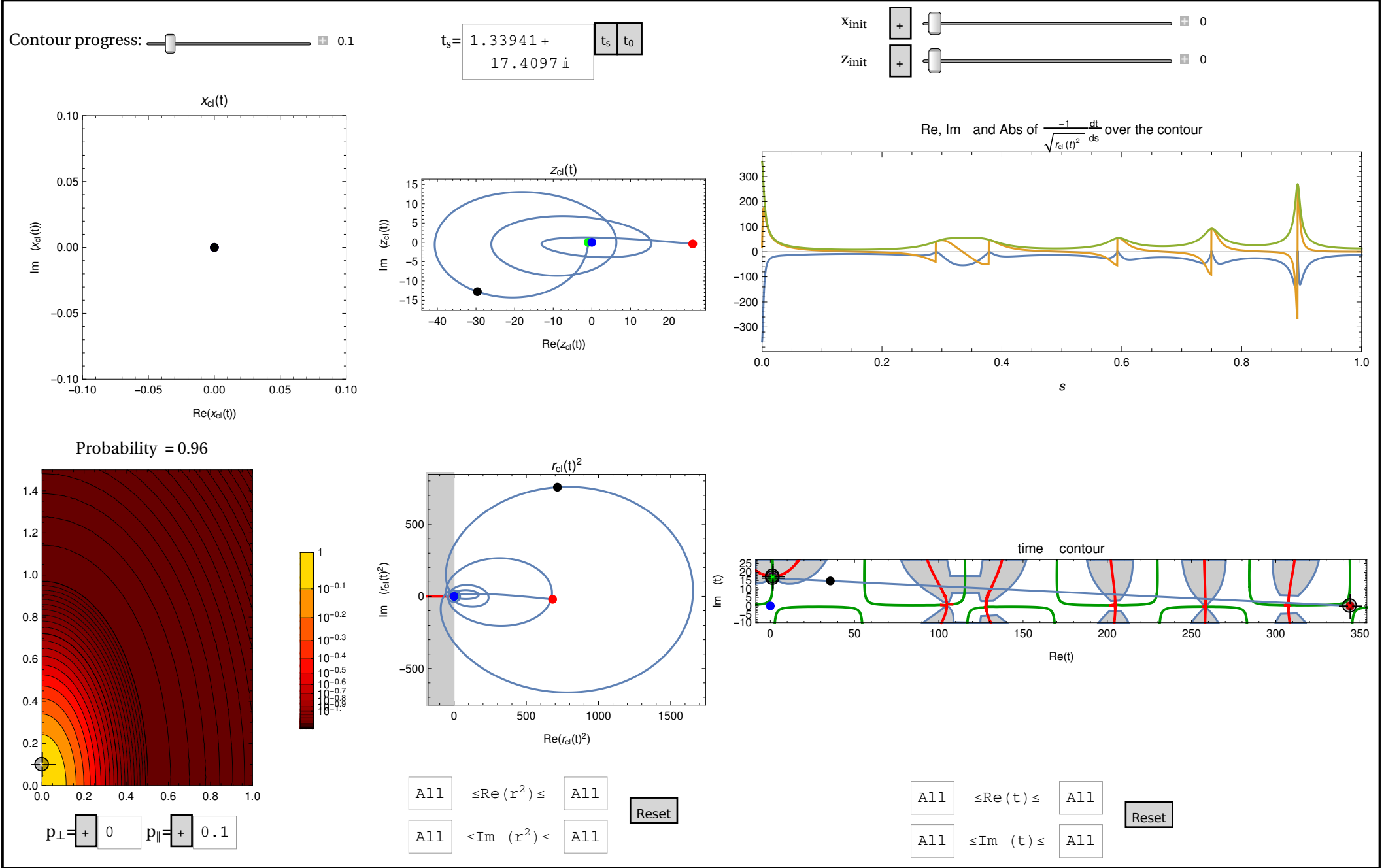
- This is very close to the peak of the ionization probability, with probability as high as 98%. This cannot be ignored.
- After about $p_{\parallel} = 0.075$, the branch cuts cross completely the real time axis. It is not clear to me, at all, that there is any way to avoid them.

During the recollision, most of the action is in the plot of $r_{cl}(t)^2$, on bottom centre. It seems the trajectory *must* loop twice around the origin of that plot, hence the two branch cut crossings. This can be seen by playing with the contour progress slider, or by adding an intermediate point to the contour path at $"tκ" + \frac{2\pi}{0.055}$.

More recollisions

Finally, branch cut crossings and loops around the zero of $r_{cl}(t)^2$ seem even more unavoidable if one takes the end of the contour a few periods further down. (Note, also, that in principle this contour should end after the pulse is finished, so this dragging-out is necessary.)


```
dashboardPlotter[{0.05, 0.055}, 1.007, {"tκ", "t0" + 3  $\frac{2\pi}{0.055}$ }, {0, 0.1}]
```

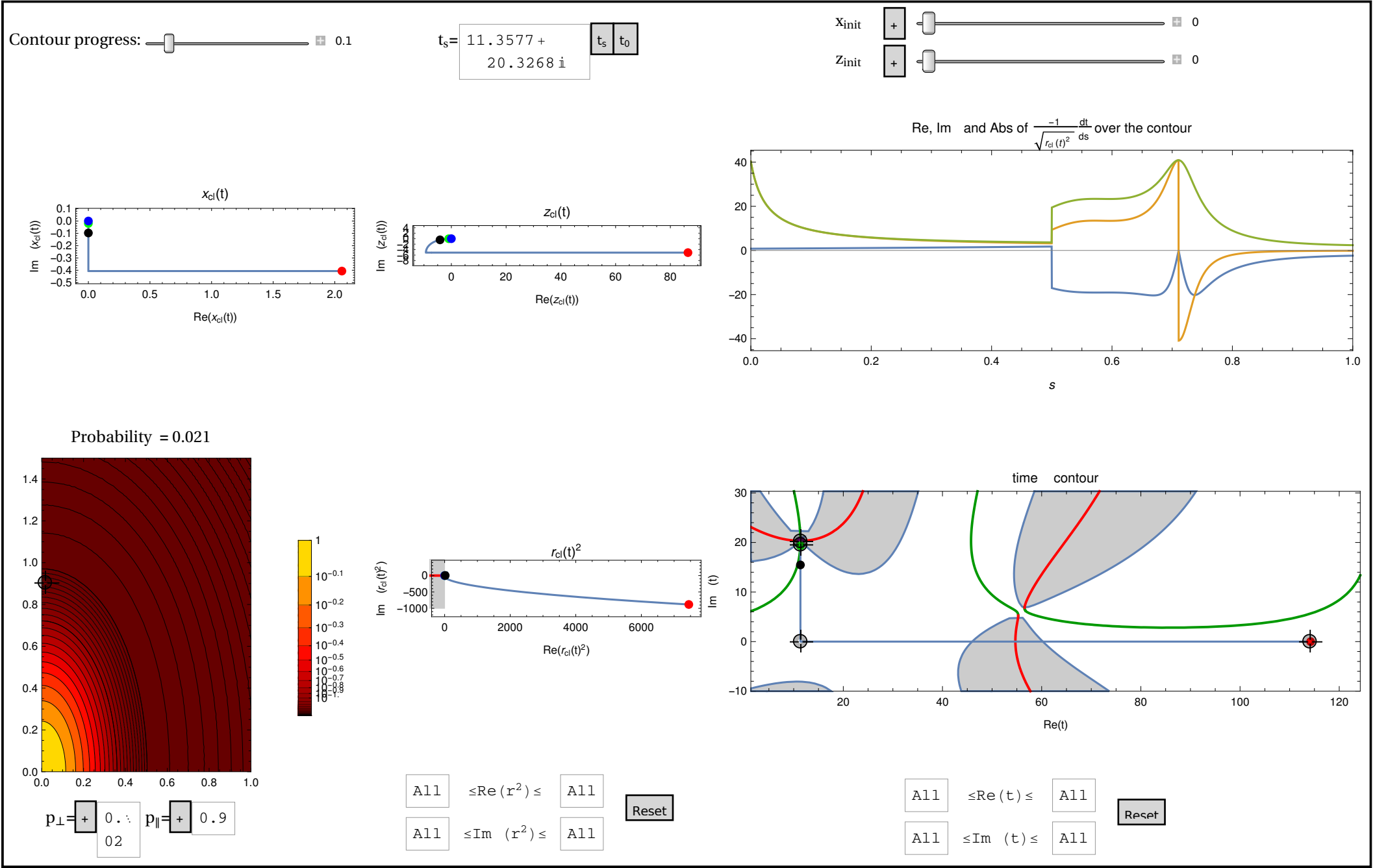


- Points of note:
- It's no longer clear to me whether the branch cuts come in pairs or not. It would be nice if they did, as crossing square-root branch cuts twice returns you to the same branch of the Riemann surface, if that even makes sense in the current context, but I don't know yet whether it's the case or not.
 - All of the branch cuts definitely do cut the real axis and at this point it's anyone's guess what to do with the time integration over the contour.
 - I really, really enjoy the multiple loops around the zero of $r_{cl}(t)^2$.

What I'm working on now

is a way to automatically choose contours that will avoid the branch cuts in situations like this,

```
dashboardPlotter[{0.05, 0.055}, 1.007, {"tκ", "t0", "T"}, {0.02, 0.9}]
```



or like this



```
Needs["EPTToolbox`", NotebookDirectory[] <> "EPTToolbox.m "]
Needs["ARMSupport`", NotebookDirectory[] <> "ARMSupport.m "]
Needs["QuODD`", NotebookDirectory[] <> "QuODD.m "]

$HistoryLength=5;

Button["Export CDF",
  Export[NotebookDirectory[] <> "Quantum Orbits Dynamic Dashboard.cdf", Import [NotebookDirectory[] <> "Quantum Orbits Dynamic Dashboard.nb"]]
]
Export[NotebookDirectory[] <> "Quantum Orbits Dynamic Dashboard.pdf", Import [NotebookDirectory[] <> "Quantum Orbits Dynamic Dashboard.nb"]];

Button["Export PDF",
  Export[NotebookDirectory[] <> "Quantum Orbits Dynamic Dashboard.pdf", Import [NotebookDirectory[] <> "Quantum Orbits Dynamic Dashboard.nb"]]
]

Export PDF
```