

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 CC BY-SA licensed due to restrictions of the format.

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.
- **Calling syntax**, to make and customize new Dashboards.
- **Some physics examples**.

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. $\boldsymbol{A}(t) = -\frac{F}{\omega} \boldsymbol{z} \sin(\omega t)$); 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

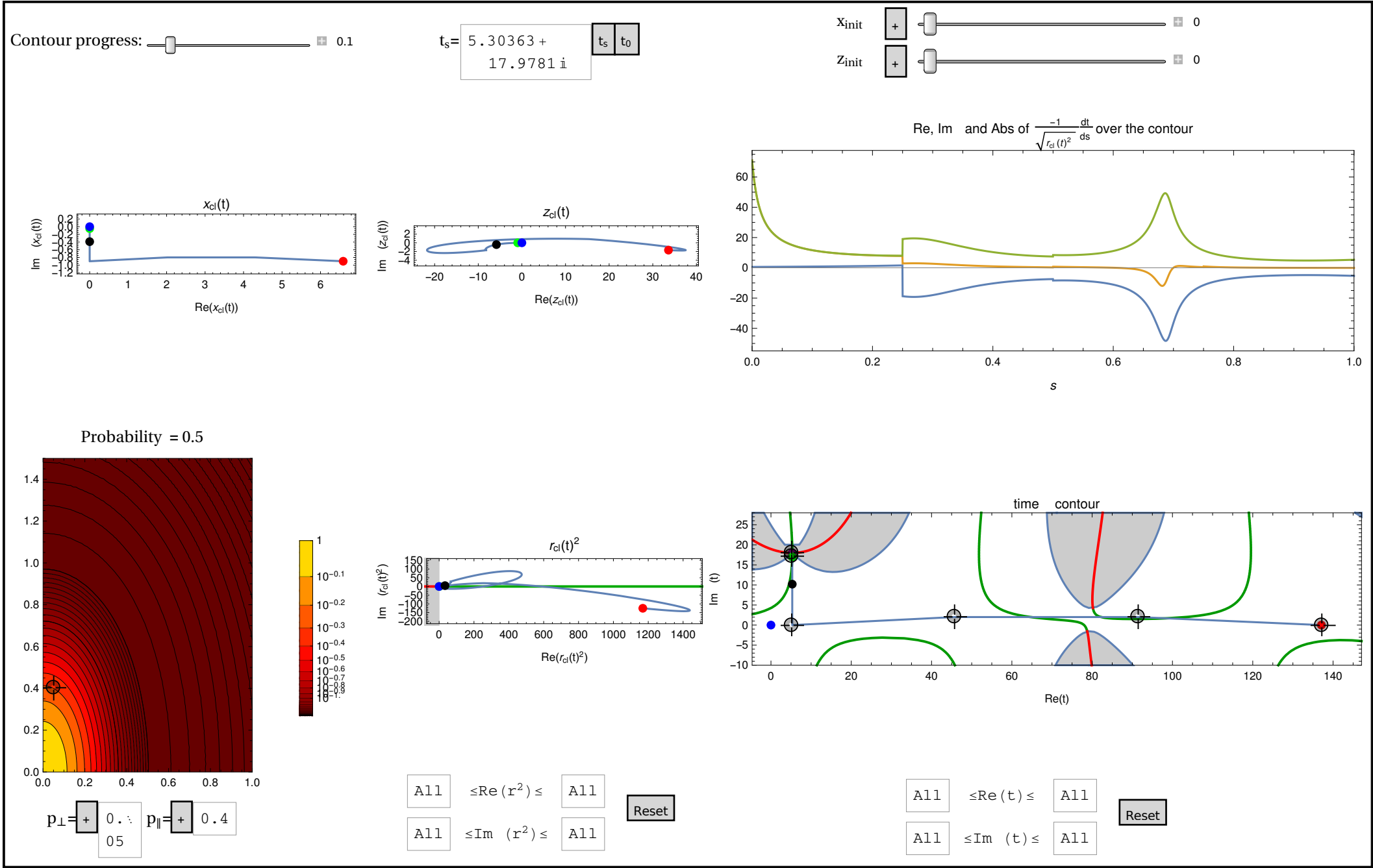
$$U(\boldsymbol{r}_{\text{cl}}(t)) = \frac{-1}{\sqrt{r_{\text{cl}}(t)^2}},$$

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:

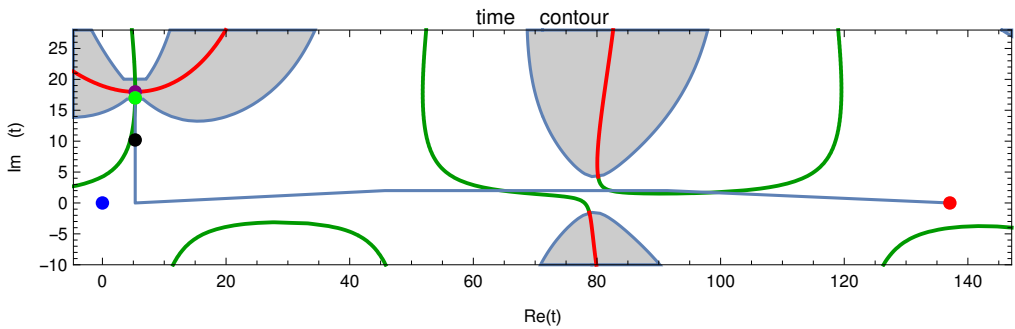
```
dashboardPlotter[{0.05, 0.055}, 1.007, {"tκ", "t0", 0.4 "T"+2 i, 0.8 "T"+2 i, 1.2  $\frac{2 \pi}{0.055}$ }, {0.05, 0.4}]
```



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 



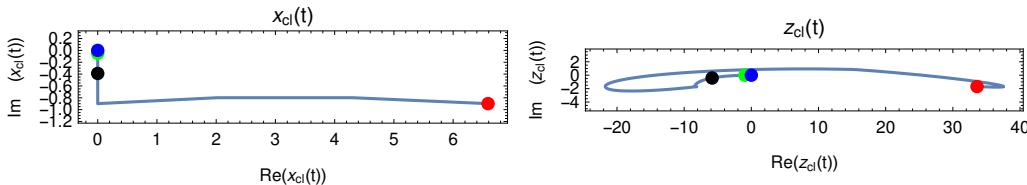
In addition, the time plane shows other relevant structures which describe the complex-valued function $r_{\text{cl}}(t)^2$:

- The branch cuts of the square root, i.e. the times for which $r_{\text{cl}}(t)^2$ is real and negative, are shown as red lines.
- By contrast, the green lines show the times for which $r_{\text{cl}}(t)^2$ 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: $\text{Re}(r_{\text{cl}}(t)^2) < 0$. 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. The start of the contour (green dot) is usually at $t_k = t_s - i/\kappa^2$, when the electron is one ground-state unit length ($1/\kappa$) away from the core.

Trajectory plots

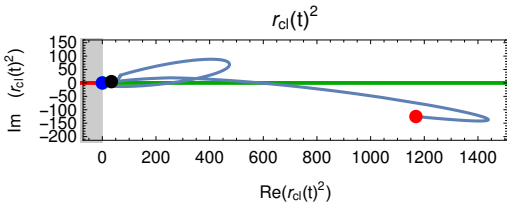
The effect of the contour is seen most clearly on the semiclassical trajectory, $r_{\text{cl}}(t)$, and this is displayed component-wise on the top left and centre. Two components are important: the x component, transverse polarization, which is a scaled copy of the time contour (since $x_{\text{cl}}(t)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



The blue line follows the contour's image on the position planes - i.e. the complex-valued trajectory, and the dots are the same as in the time plane.


Position-squared plot

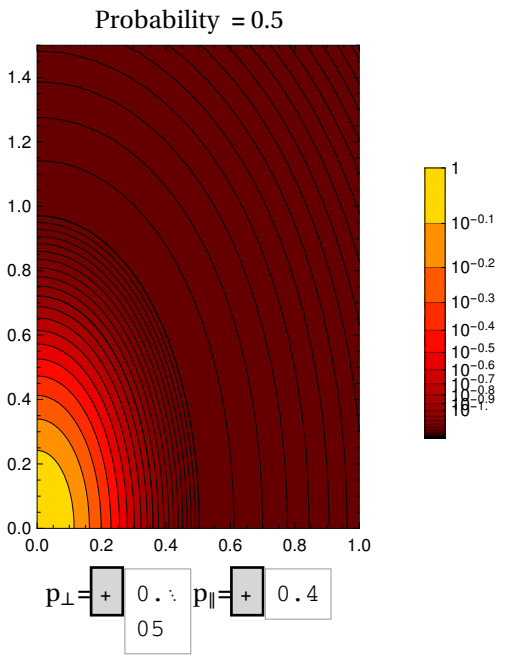
The plot at bottom centre is exactly analogous but it plots $r_{\text{cl}}(t)^2 = x_{\text{cl}}(t)^2 + z_{\text{cl}}(t)^2$ on its complex plane. This squared position is relevant because it is the argument of the square root in the Coulomb potential $U(r_{\text{cl}}(t)) = -1/\sqrt{r_{\text{cl}}(t)^2}$, so it is precisely when $r_{\text{cl}}(t)^2$ is real and negative that the square root changes sign at its branch cut.



This plot therefore has a red line along its negative real axis, which the contour should never cross. Similarly, a green line marks the positive real axis, and the gray region is where $\text{Re}(r_{\text{cl}}(t)^2) < 0$. The lines and gray regions in the time plane are exactly the pre-images of these lines and regions via the complex map $t \mapsto r_{\text{cl}}(t)^2$.

Momentum plane

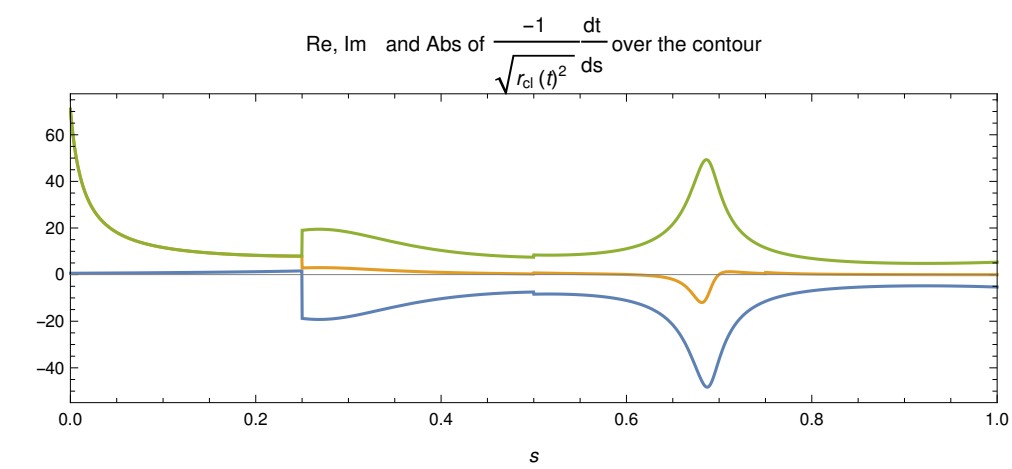
The other important control is on the momentum plane, which controls the final asymptotic momentum \mathbf{p} of the electron. The control is at bottom left and shows the first quadrant of the $(p_x, p_z) = (p_{\perp}, p_{\parallel})$ plane, with input underneath for finer control and to change quadrant. The momentum can be controlled using the selector marked 



The background shows the SFA ionization probability $e^{-2\text{Im}\left(\int_b^t \left[l_p + \frac{1}{2}(\mathbf{p} + \mathbf{A}(\tau))^2\right] d\tau\right)}$, with linear colour scale and logarithmic contours, and normalized with respect to its value at the peak. (Tooltips give the precise contour values.) The ARM ionization probability can be substantially modified from these values but this is a good guide to which momenta are experimentally relevant and which ones are not.








Coulomb Integrand

Finally, the graph at top right shows the Coulomb integrand, $U(r_{\text{cl}}(t)) = -1 / \sqrt{r_{\text{cl}}(t)^2}$. The horizontal axis is a normalized contour variable s , and the integrand has been multiplied by the derivative $\frac{dt}{ds}$ so that the integral is invariant. This causes apparent discontinuities at the contour's corners, where different legs - with different lengths ad therefore different derivatives $\frac{dt}{ds}$ - are joined together.



That behavior is distinct from the sign discontinuity in this integrand whenever a branch cut is crossed.

Controlling the Dashboard


- The Dashboard can be interacted with in several ways:
 - 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 $r_{\text{cl}}(t) = \int_{t_s}^t (\boldsymbol{p} + \boldsymbol{A}(\tau)) \, d\tau$ starts, can be controlled using the input box at top centre. Clicking the  button sets it to the saddle point time (which obeys $\frac{1}{2} (\boldsymbol{p} + \boldsymbol{A}(t_s))^2 + I_p = 0$), 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, $r_{\text{cl}}(t) = r_{\text{init}} + \int_{t_s}^t (\boldsymbol{p} + \boldsymbol{A}(\tau)) \, d\tau$. This is required if the start time is set to zero, as the trajectory is meant to start on the ARM boundary.

Calling syntax

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`

dynamicDashboardPlotter [[F, ω], κ] plots a dashboard for field amplitude F at frequency ω , for ionization potential $\kappa^2/2$.
dynamicDashboardPlotter [[F, ω], κ , path] institutes the desired path, where the strings "t κ ", "ts", "t0" and "t" will be replaced by the corresponding functions of momentum , and "T" is a laser period . Default is {"t κ ", "t0", "T"}.
dynamicDashboardPlotter [[F, ω], κ , path, {point , ppinit }] specifies initial values of point and pp init for p _x and p _y .

Entering +k will pull up these templates for easier input.

Before making the dashboard, make sure the packages EPToolbox`, ARMSupport` and QuODD` are correctly loaded by running the following initialization cell.

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

It is also helpful to reduce the \$HistoryLength of the notebook, to prevent the cache from storing old Dashboards.

```
$HistoryLength= 5;
```

To modify any function, find it in QuODD.nb and modify it. To see the available functions and variables, use

```
Names ["QuODD`*"]

{colourScale, contourProgressController, dashboardPlotter, ionizationProbabilityColorFunction
 ionizationProbabilityPlqtmomentumPlaneControls , rangeReset, rInitController, statifyDashboard, timeContours ,
 timeIntegrandPlotter , timePathPlotter , trajectoryPlotter, tsController, $dashboardMainSize $largeBlockSize, $smallBlockSize }
```

Use the same syntax for the ARMSupport` and EPToolbox packages. To see the documentation for any function in either package, use the ?function syntax as above.

Some physics examples

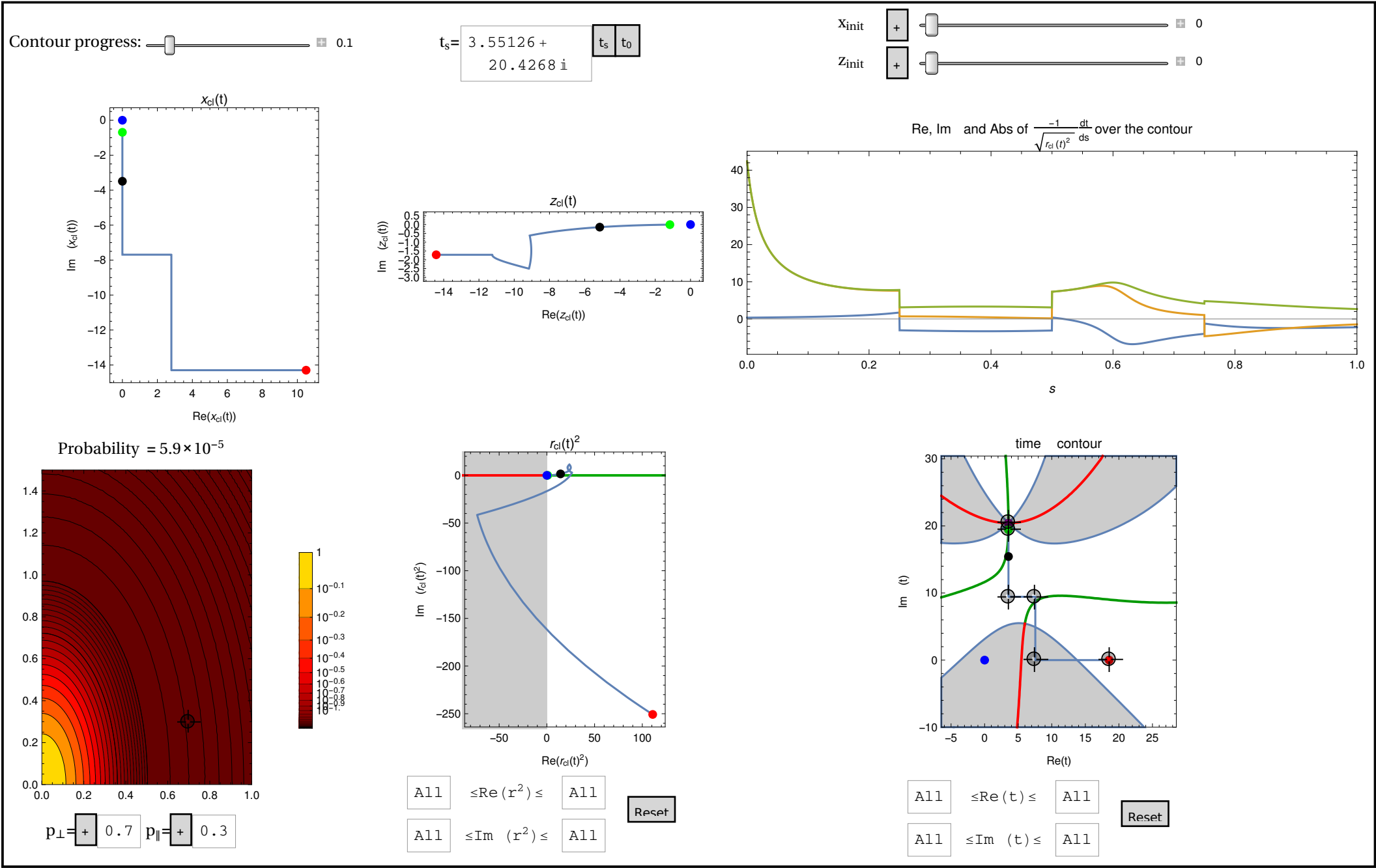
This section explores some examples where the Dashboard is a useful visualization tool.

The ‘standard’ contour can fail

The integration contour is usually taken from the ionization time t_s (or the time $t_\kappa = t_s - i / \kappa^2$ at which it is $1 / \kappa$ away from the ion), directly down to the real axis at $t_0 = \text{Re}(t_s)$, and then along the real axis until the (large) detection time T . This contour usually works well enough but there cases do exist within the first-principles ARM theory where it leads to incorrect results.

One neatly-put situation where this can happen is when the transverse momentum is very large. In this case, during the ionization step the transverse coordinate $x_{\text{cl}}(t_0) = p_x(t_0 - t_s) = -i \tau_T p_x$ can accumulate a large imaginary part, because the time interval is imaginary and the transverse velocity $v_x = p_x$ is real. In these cases, a large imaginary transverse position causes a mostly negative squared position $r_{\text{cl}}(t)^2 = x_{\text{cl}}(t)^2 + z_{\text{cl}}(t)^2$, which can then cross a branch cut of the square root.

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



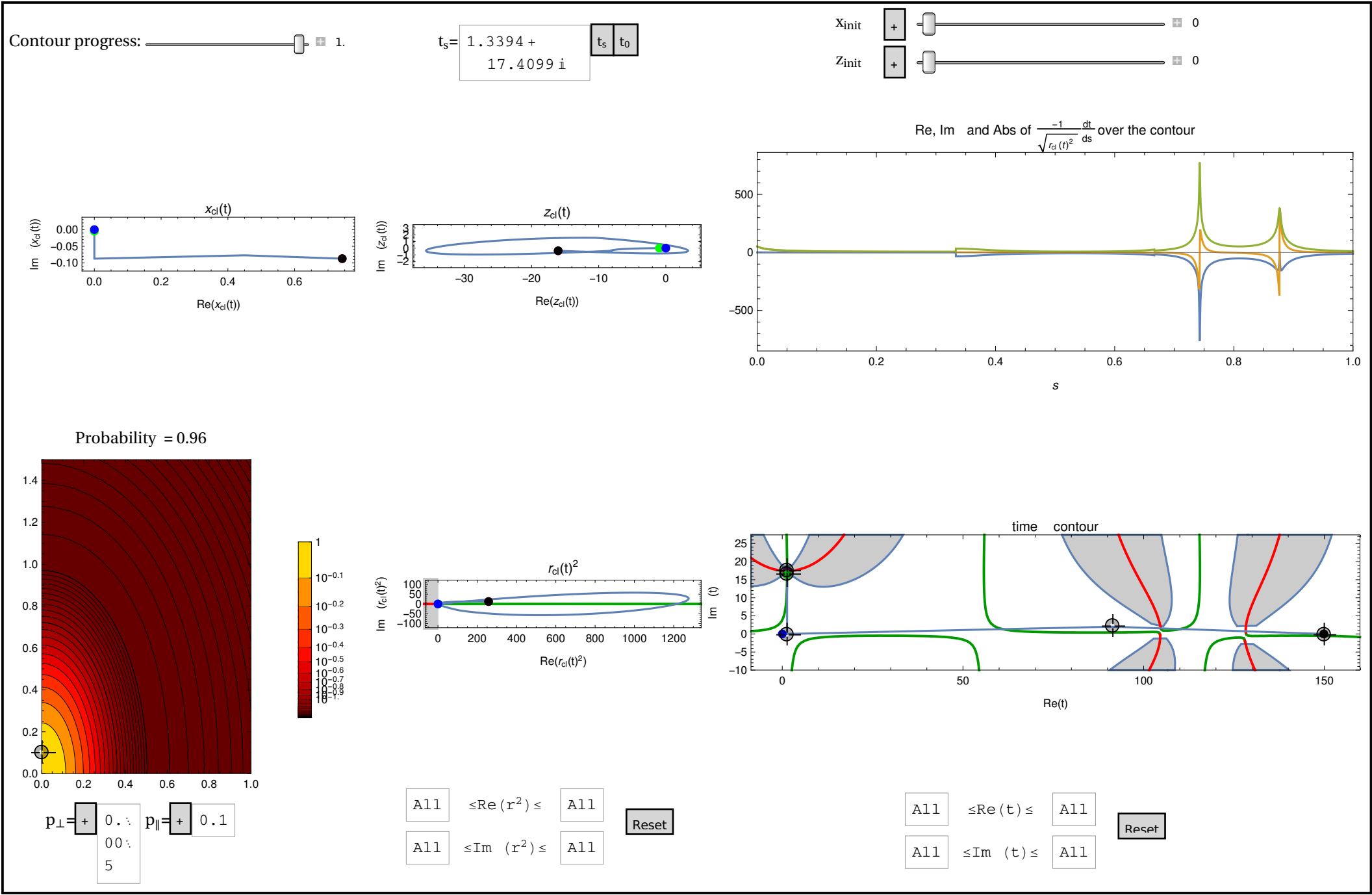
In this situation, the contour must go to the right at some point before it reaches the real axis, and go down after the branch cut has been avoided. This could be interpreted as a 'delay' in the ionization time (which could be defined as the earliest real time from which a real-time trajectory is a good model), though this situation should be treated carefully.

In particular, transverse momenta large enough to cause this effect are generally too large to be experimentally relevant, and the effect occurs in the wings of the distribution. Nevertheless, the effect is a possibility and should be kept in mind. As explored below and elsewhere, other situations with higher probabilities also incur branch cuts which cross the real axis.

Recolliding electrons

Recolliding electrons look quite interesting with these tools. In particular, if the electron approaches the ion then the real part of the trajectory, $\text{Re}(r_{cl}(t))$, becomes very small. This means that its imaginary part $\text{Im}(r_{cl}(t))$ will dominate the position, making the square $r_{cl}(t)^2$ predominantly negative and forcing the presence of a branch cut.

dashboardPlotter[{0.05, 0.055}, 1.007, {"tκ", "t0", $0.8 \frac{2\pi}{0.055} + 2 i$, "t0"+ $1.3 \frac{2\pi}{0.055}$ }, {0.005, 0.1}]



Such branch cuts tend to come in pairs, for trajectories which pass the origin, turn around, and pass the origin again on the way back. However, the total number of relevant branch cuts will in general be odd.

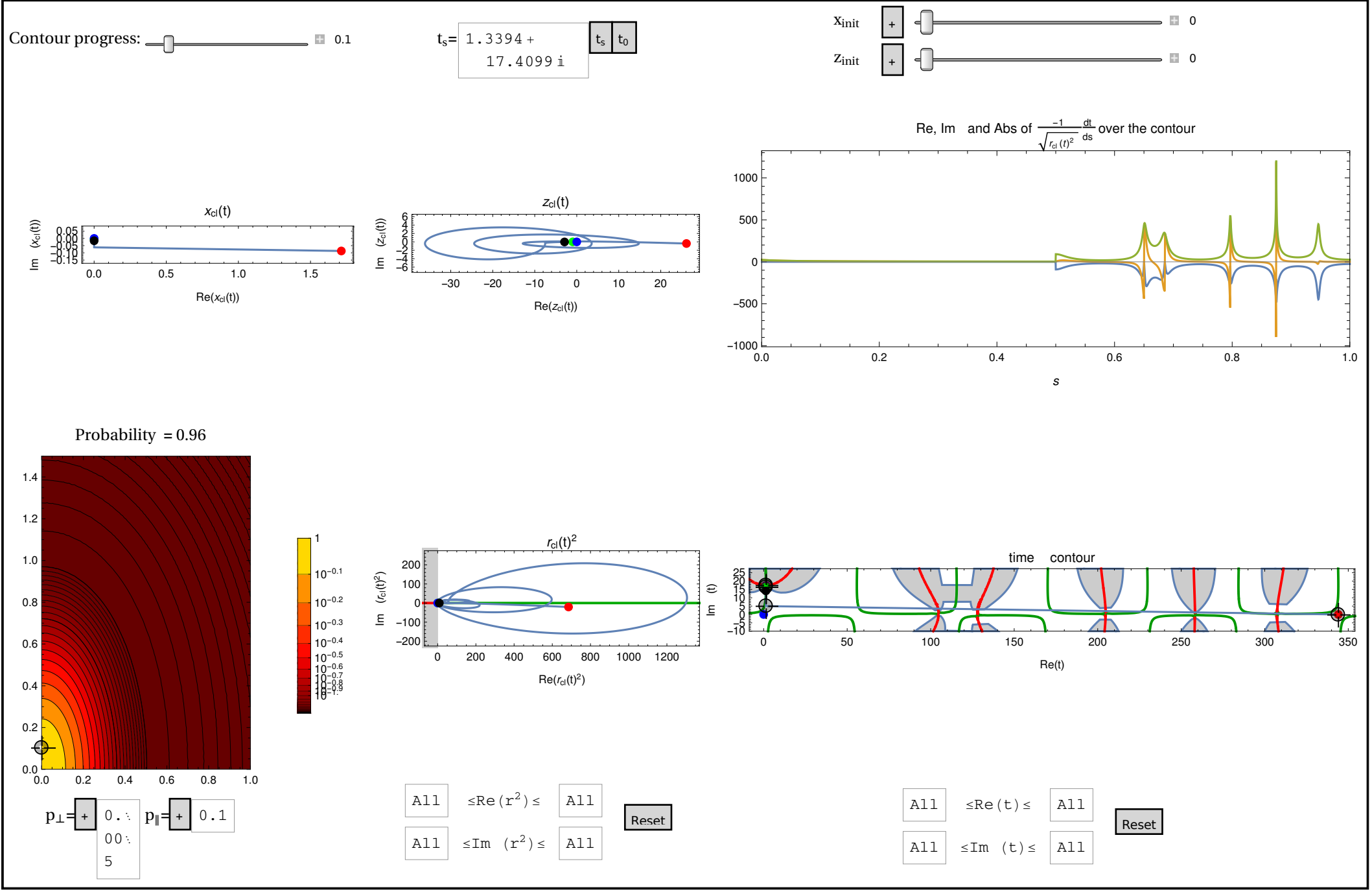
It's important to note that these events are quite close to the maximum of the SFA photoelectron distribution and they are therefore relevant for any semiclassical theory. Recollisions must be treated carefully in semiclassical theories and these tools help identify their positions within the complex-valued formalism.

Does an approach count as a recollision if the real part of the position is small but the imaginary part is not? If changing the contour changes the balance between the real and imaginary parts of the position, which contour is optimal? As we show in Ref. [1], in these cases there is always a special point in between each 'slalom gate' of branch cuts. (This point is a saddle point of $r_{cl}(t)^2$, and therefore a solution of the closest-approach equation $r_{cl}(t) \cdot \mathbf{v}(t) = 0$.) Contours which pass through this point will minimize the imaginary position at the crucial recollision time, trading a real position for a slightly imaginary time.

More recollisions

If one looks at a relatively low energy electron over multiple laser periods, the real part may revisit the origin several times and this causes a correspondingly larger set of branch cuts to be avoided.


```
dashboardPlotter[{{0.05, 0.055}, 1.007, {"tκ", "t0"+5 i, "t0"+3  $\frac{2 \pi}{0.055}$ }, {0.005, 0.1}]
```



dashboardPlotter[{0.05, 0.055}, 1.007, {105, 110, 115, 120, 125}, {0.001, 0.071}]

