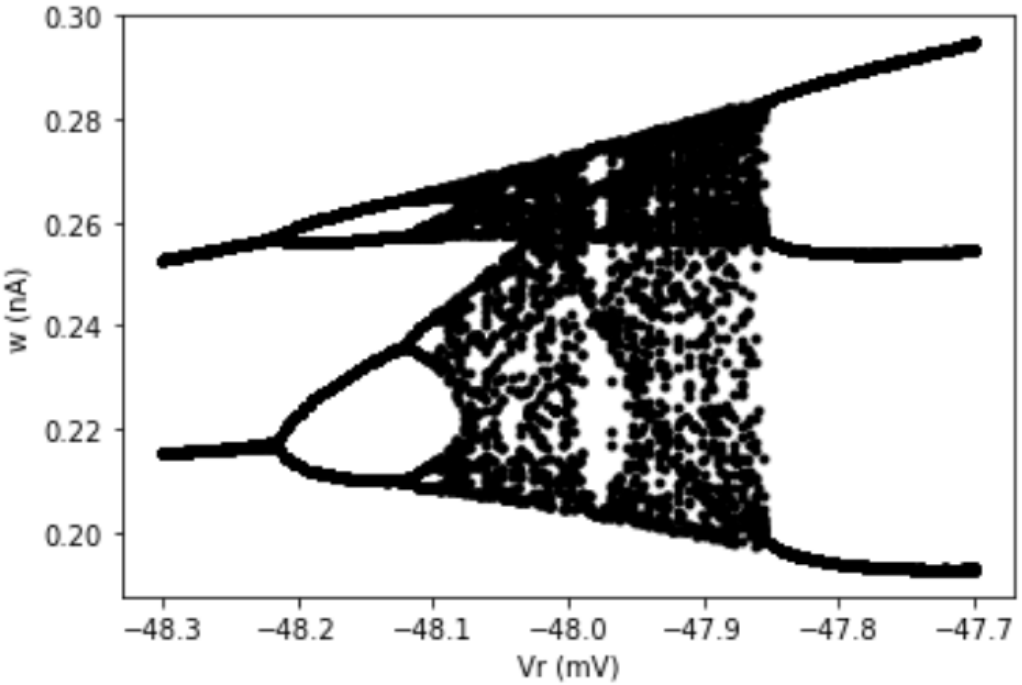
Chaos in the AdEx model

Blokhnin Kirill, Shenzhen MSU-BIT UNIVERSITY
2017

Fig. 8B from: Touboul, J. and Brette, R. (2008). Dynamics and bifurcations of the adaptive exponential integrate-and-fire model. Biological Cybernetics 99(4-5):319-34.

- This shows the bifurcation structure when the reset value is varied (vertical axis shows the values of w at spike times for a given a reset value Vr).

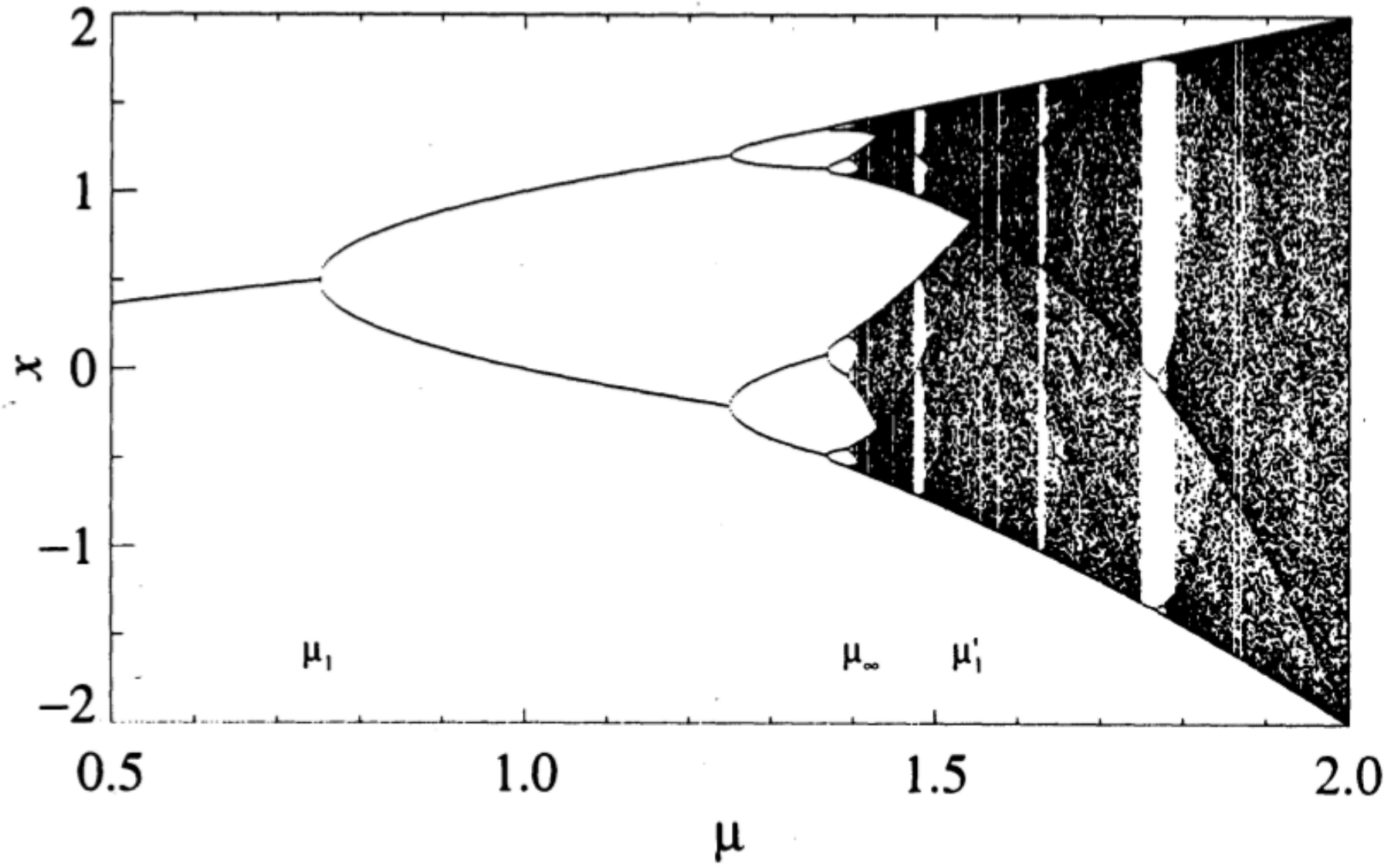


The Integrate-and-Fire Model

The integrate-and-fire neuron is one of the simplest models of a neuron's electrical properties and probably the most commonly used in the field of neuroscience. The essence of the model is to divide the voltage changes of the neuron into two parts:

- 1) Below threshold, it is assumed that the membrane behaves passively (i.e. has no voltage-dependent ion channels) and acts as a leaky capacitor whose voltage, in the absence of injected current, decays (or "leaks") to a resting level E L (short for "E Leak ").
- 2) When the voltage reaches the action potential threshold (due to injected currents charging up the membrane), the model assumes that the voltage spikes immediately to a level V spike and is then immediately reset to a hyperpolarized level V reset . There is no explicit modeling of the ion channel kinetics responsible for this spiking. Rather, it is simply assumed (a reasonable assumption...) that once the cell reaches its threshold it will rapidly produce an action potential and reset itself. The reason we can get away with this assumption is that we don't really care about the exact shape of the action potential: since all action potentials sent down the axon are to a good approximation identical, the only informative feature of a neuron's spiking is the times at which the action potentials occur

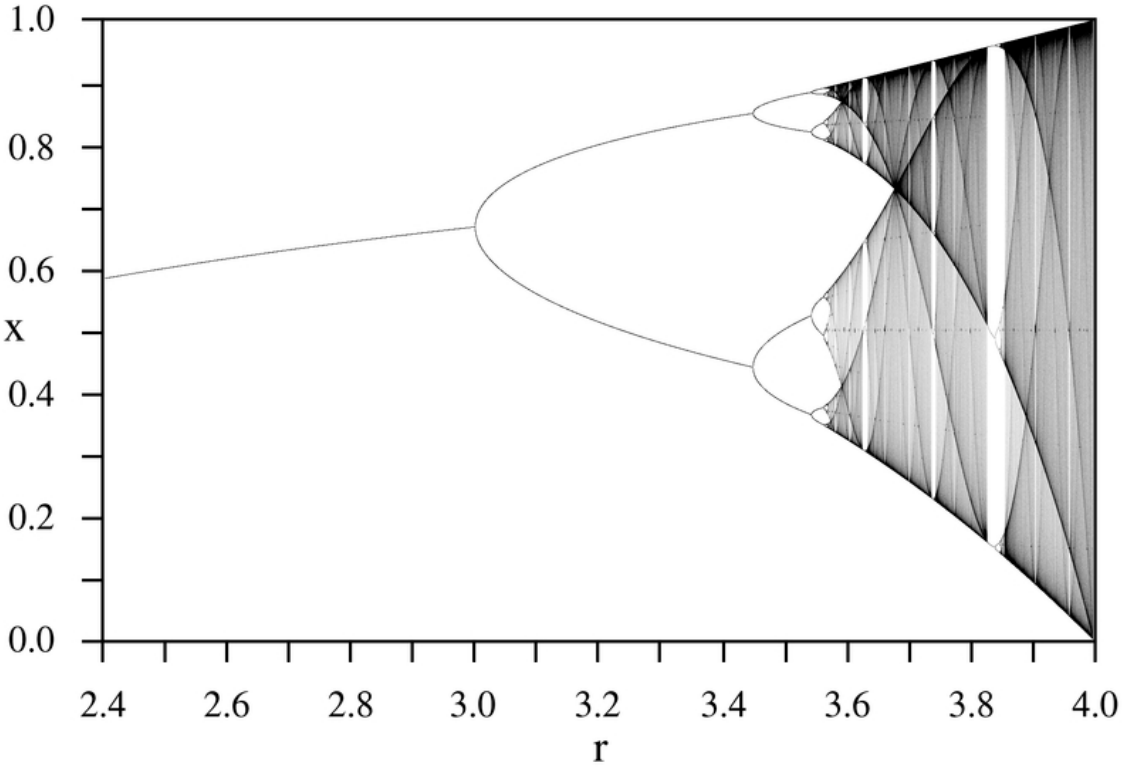
PURE CHAOS



An example is the bifurcation diagram of the logistic map:

$$x_{n+1} = r x_n (1 - x_n).$$

The bifurcation parameter r is shown on the horizontal axis of the plot and the vertical axis shows the set of values of the logistic function visited asymptotically from almost all initial conditions.



The bifurcation diagram shows the forking of the periods of stable orbits from 1 to 2 to 4 to 8 etc. Each of these bifurcation points is a period-doubling bifurcation. The ratio of the lengths of successive intervals between values of r for which bifurcation occurs converges to the first Feigenbaum constant.

The diagram also shows period doublings from 3 to 6 to 12 etc., from 5 to 10 to 20 etc., and so forth.

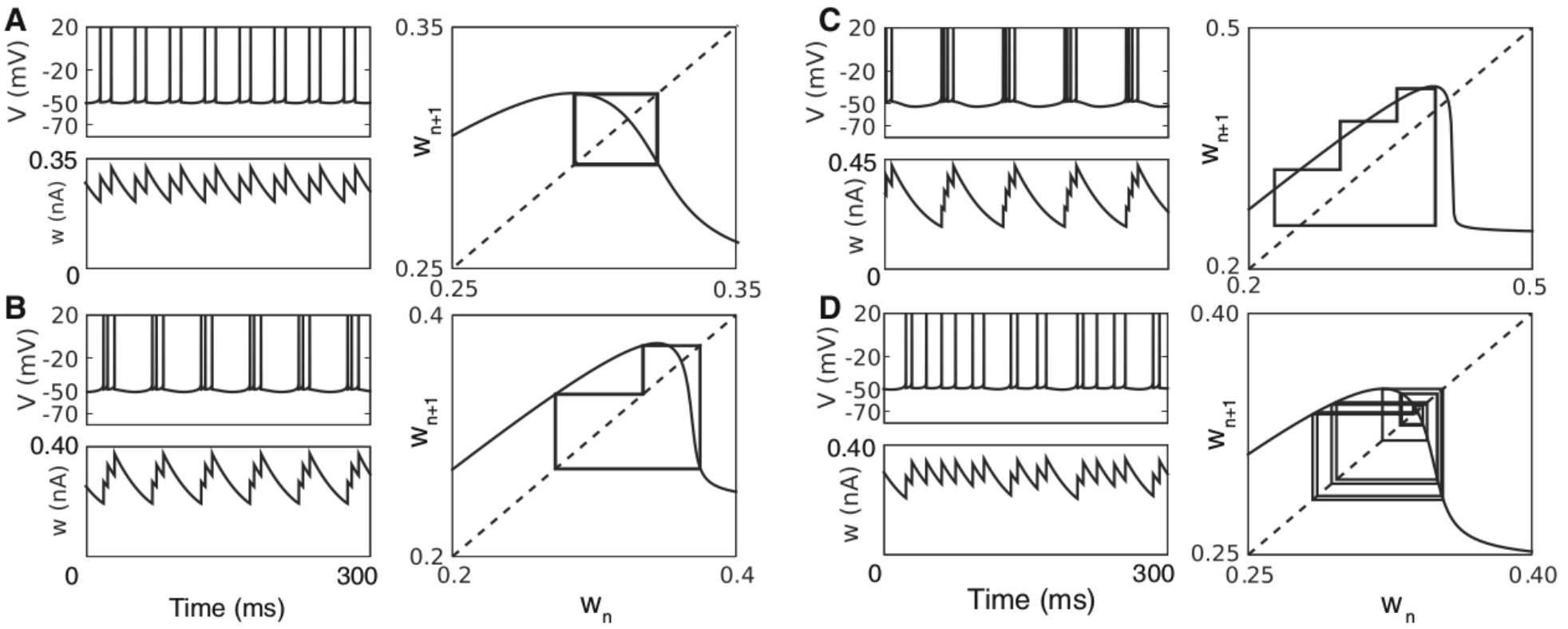


Fig. 7 Bursting and chaos. Each *panel* shows a sample response (V and w) from the model, with different values of V_r (parameters: $C = 281$ pF, $g_L = 30$ nS, $E_L = -70.6$ mV, $V_T = -50.4$ mV, $\Delta_T = 2$ mV, $\tau_w = 40$ ms, $a = 4$ nS, $b = 0.08$ nA, $I = 0.8$ nA). A burst with n burst occurs in the decreasing part of Φ, inducing a slower trajectory. **a** Bursting with two spikes ($V_r = -48.5$ mV). **b** Bursting with three spikes ($V_r = -47.7$ mV). **c** Bursting with four spikes ($V_r = -47.2$ mV). **d** Chaotic spiking ($V_r = -48$ mV)

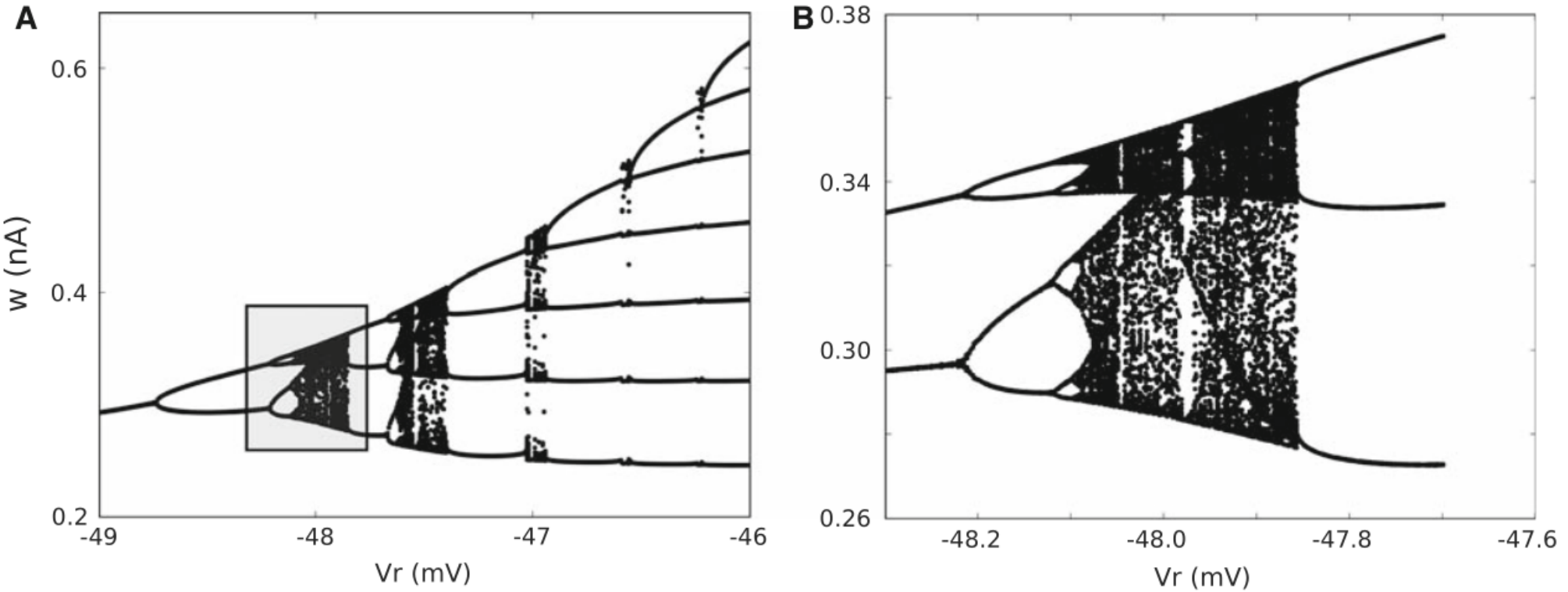


Fig. 8 Bifurcation structure with increasing V_r (same parameters as in Fig. 7). **a** Bifurcation diagram showing a period adding structure (orbits under the adaptation map Φ with varying values for V_r). *Fixed points* indicate regular spiking, *periodic orbits* indicate bursting, *dense orbits* indicate chaos. **b** Zoom on the bifurcation diagram **a** (as indicated by the *shaded box*), showing a period doubling structure

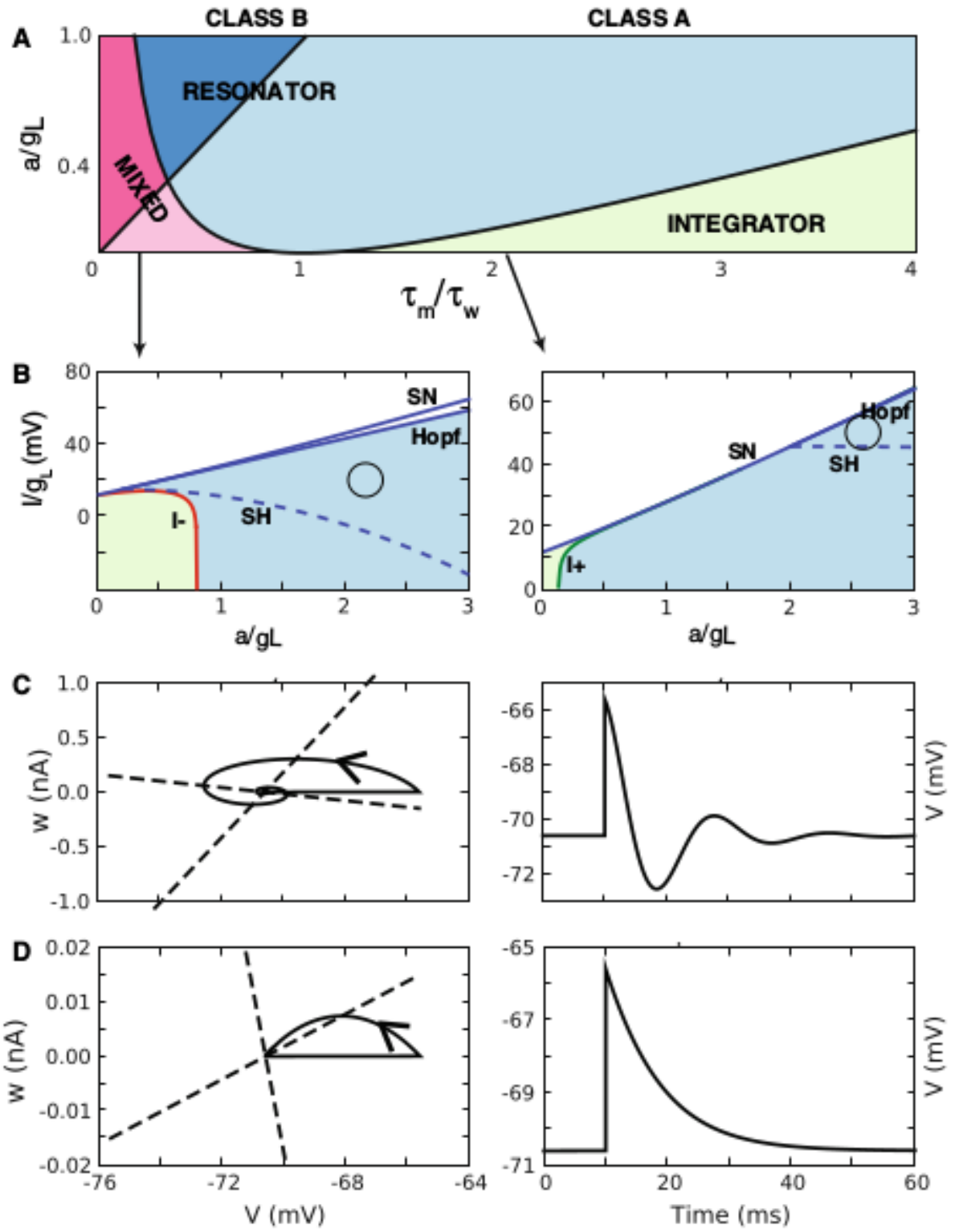


Fig. 3 Oscillations. **a** Behavior of the model as a function of a/g_L and τ_m/τ_w. *Light* (*dark*) *colors* indicate class A (class B) parameters. *Blue* indicate resonator mode (oscillations for any or almost any I). *Green* indicate integrator mode (oscillations for any I). *Pink* indicate mixed mode (resonator if I is large enough, otherwise integrator). **b** Behavior of the model as a function of a/g_L and I/g_L for $\tau_m = 0.2\tau_w$ (*left*) and $\tau_m = 2\tau_w$ (*right*). *White* indicate spiking; *blue* indicate oscillations; *green* indicate no oscillation. Spiking occurs when I is above the saddle-node curve (*SN*) in the class A regime, and above the Hopf curve (Hopf) in the *class B* regime. A repulsive limit cycle (*circle*) exists when I is above the saddle-homoclinic curve (*SH*; only for *class B*). Oscillations occur when $I_- < I < I_+$ (on the *left*, $I_+ \geq I_{SN}$; on the *right*, $I_- = -\infty$). **c**, **d** Response of the system to a short current pulse (Dirac) near the resting point, in the resonator regime (**c** $a = 10g_L$, $\tau_m = \tau_w$) and in the integrator regime (**d** $a = 0.1g_L$, $\tau_m = 2\tau_w$). *Left* response in the phase space (V, w); *right* voltage response in time

In [1]:
```
from brian2 import *
```

In [2]:
```
#setting the clock
defaultclock.dt = 0.01*ms
```

In [3]:
```
#inserting system information
C = 281*pF
gL = 30*nS
EL = -70.6*mV
VT = -50.4*mV #
DeltaT = 2*mV
tauw = 40*ms ##
a = 4*nS ##
b = 0.08*nA ##
I =.8*nA
Vcut = VT + 5 * DeltaT  # practical threshold condition
N = 200
```

In [4]:
```
#inserting nessesary equations
eqs = """
dvm/dt=(gL*(EL-vm)+gL*DeltaT*exp((vm-VT)/DeltaT)+I-w)/C : volt
dw/dt=(a*(vm-EL)-w)/tauw : amp
Vr:volt
"""
```

In [5]:
```
#setting the neuronal group
neuron = NeuronGroup(N, model=eqs, threshold='vm > Vcut',
                     reset="vm = Vr; w += b", method='euler')
neuron.vm = EL
neuron.w = a * (neuron.vm - EL)
neuron.Vr = linspace(-48.3 * mV, -47.7 * mV, N)  # bifurcation parameter
```

In [6]:
```
#starting the simulation
init_time = 3*second
run(init_time, report='text')  # we discard the first spikes
```

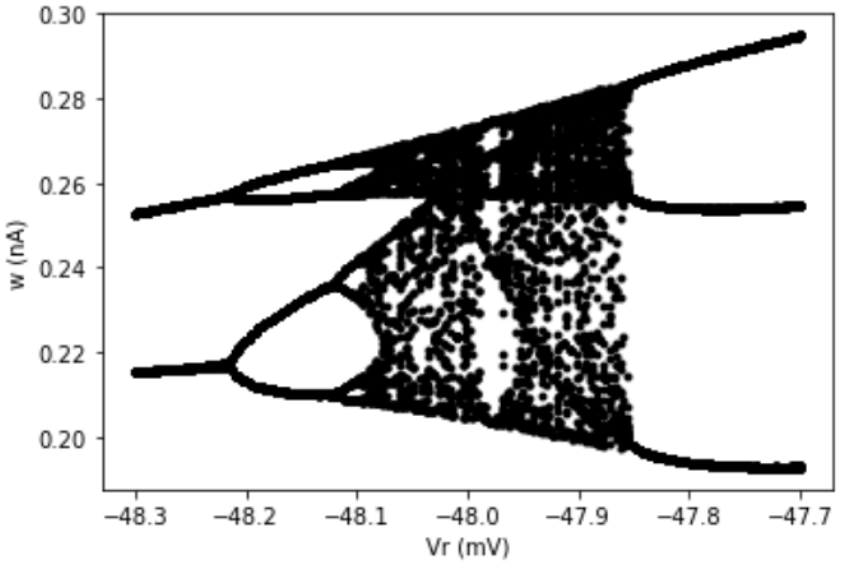
```
Starting simulation at t=0. s for a duration of 3. s
3. s (100%) simulated in 3s
```

In [7]:
```
#adding the spike monitor
states = StateMonitor(neuron, "w", record=True, when='start')
spikes = SpikeMonitor(neuron)
run(1 * second, report='text')
```

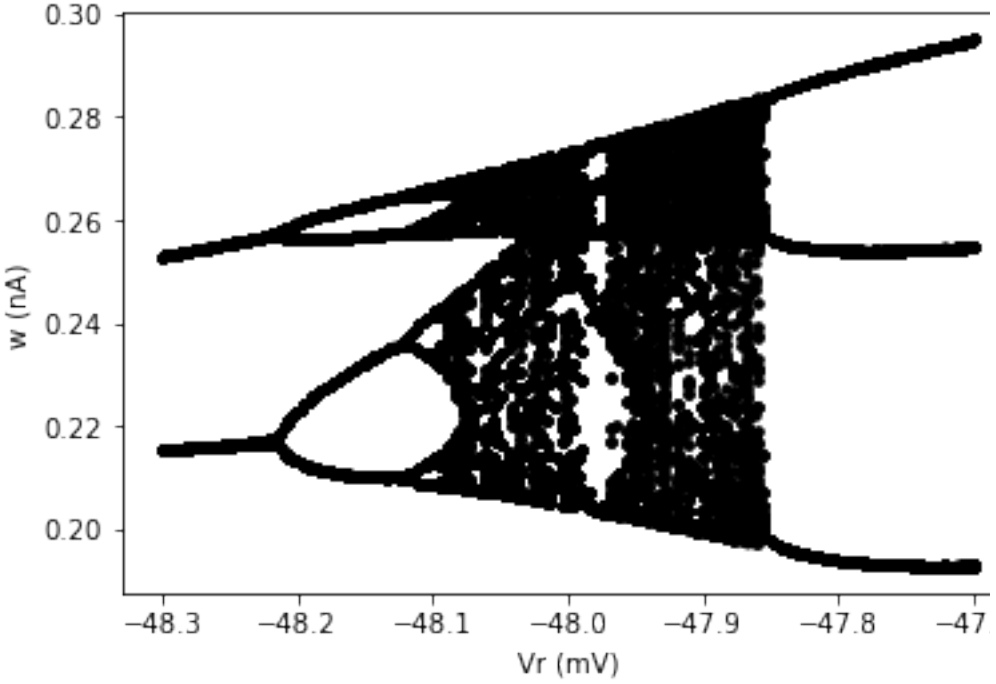
```
Starting simulation at t=3. s for a duration of 1. s
1. s (100%) simulated in 2s
```

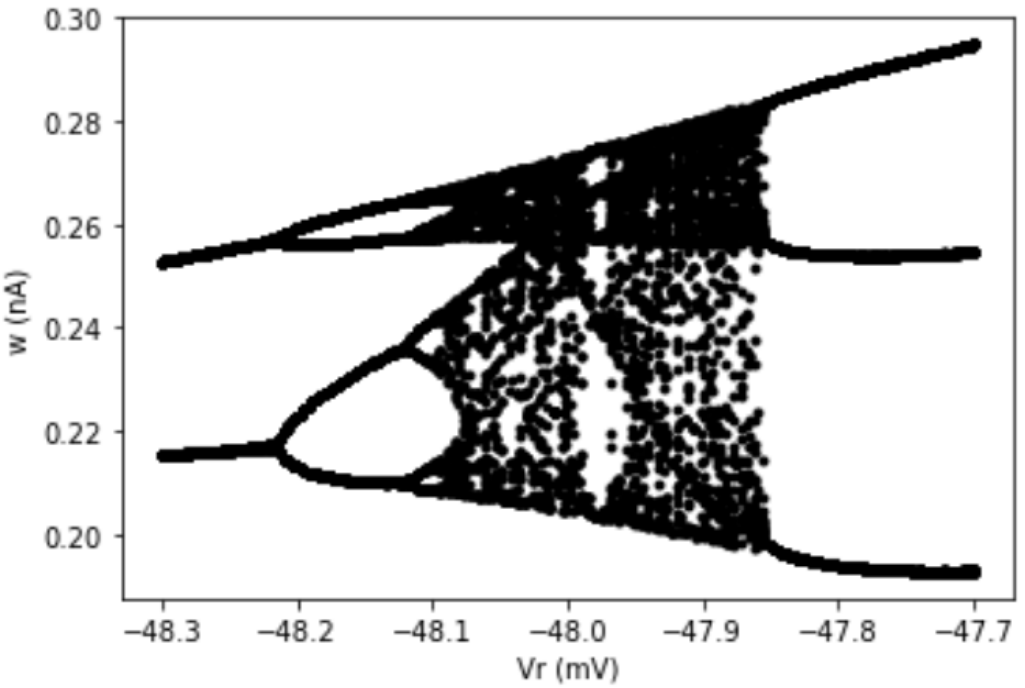
In [8]:
```
# Get the values of Vr and w for each spike
Vr = neuron.Vr[spikes.i]
w = states.w[spikes.i, int_((spikes.t-init_time)/defaultclock.dt)]
```

In [9]:
```
#plotting the graph
figure()
plot(Vr / mV, w / nA, '.k')
xlabel('Vr (mV)')
ylabel('w (nA)')
show()
```



Attachment of a Spike monitor for 9 sec. and 3 sec.





The simulation graph with tauw=50 and tauw=60