Duman et al, http://www.jgp.org/cgi/doi/10.1085/jgp.200709915

Provenance of PC12 Cells

The PC12 D19 cells came from Dr. Sandra Bajjalieh (University of Washington) who obtained them from Dr. Thomas F.J. Martin (University of Wisconsin). The D19 strain had been selected much earlier by subcloning in the Martin lab for numerous secretory granules and for robust depolarization-induced calcium responses and was used in all exocytosis studies of the Martin lab. The PC12 cell stock came to Wisconsin from Dr. Erik Schwietzer and the Regis Kelly lab at UCSF where they had been conditioned to grow on supplemented DMEM. We thank Dr. Martin for tracing this history for us.

Computer Program for Local Depletion Calculations

```
//Program written for IGOR Pro (WaveMetrics, Lake Oswego, OR)
#pragma rtGlobals=1             // Use modern global access method.
Function DoItAll()  //This function runs all caculations
        PCheck()
        PDiffuse()
end

Function pCheck()  //set up arrays but stop before integrating
        PlanarArrays()
        FirstpFlux()
end

//*****************************************************************************
***************
//----------------calculate diffusion regimes for a one dimensional planar problem
//  December 18, 2007 B.H.
Function PlanarArrays ()            //defines waves and initializes values

        variable /g  plim = 50  //length of waves (number of compartments)
        Make /O/N = (plim)   dist,  pflux, ptheory, topBuff, pBflux
        Make /O/D/ N = (plim)  pcCa, pMCa, pcCaBuff, pMCaBuff          //note double precision
        ptheory = 0
        Variable/g dx = 0.02    //thickness in um per compartment
        variable cnt
        //variable MolecsPeruM = 6.022e23*1.e-6*1.e-15
        //print MolecsPeruM //     answer is   602 molecules per cubic um in a 1 uM solution
        variable pCaInit =1             //starting concentration, units uM
        pcCa = pCaInit                  //initialize cytoplasmic Ca
        //pcCa[0] = 10000               //implement this when checking instantaneous point source
        dist = dx/2                     //distance from origin (origin = membrane)
        dist += p*dx
        cnt = 0
        pMCa = pcCa * dx                //volume assumes 1 um by 1 um square cylinder
        //pMCa is molar quantity in compartment (units umol/1.e15 = 1.e-21 mol)
        variable /g  BuffFactor = 300           //Ca binding ratio, kappa
        pMCaBuff = BuffFactor* pMCa       //equilibrate Ca-bound buffer moles
```

```
	pcCaBuff = pMCaBuff/dx		//Ca-bound buffer concentration

	variable /g StartMass
	StartMass = sum (pmca) + sum (pmcabuff)		//total free and bound calcium in system
end

Function FirstpFlux()			//calculates fluxes for first step to check step size
	WAVE  dist, pcCa, pMCa, pflux, ptheory, pcCaBuff, pMCaBuff, topBuff, pBFlux
	variable /g  plim			//number of compartments
	variable/g D = 0.3e3		//diff coefficient in um2/s.  Note: 1.e-5 cm2/s = 1000 um2/s
	variable/g pdt = 0.2e-7		//small time step of integration in sec
	//for 0.02 um compartment widths, pdt can be as high as 200 ns
	//above that the calculation is not stable
	variable /g dx
	variable cnt = 0

	do
		pFlux[cnt] = -pdt * D * (pcCa[cnt] - pcCa[cnt+1]) /dx		//assume 1 um2 area
		Cnt += 1

	while (cnt < plim)
	pFlux[plim] = 0
end
//-----------------------------------------------------------integrate diffusion in time
Function PDiffuse()		//Big repetitive integration of diffusion (Euler method)
	WAVE  dist, pcCa, pMCa, pflux, ptheory, pcCaBuff, pMCaBuff, ToPBuff, pBFlux
	variable /g  plim		//number of compartments
	variable/g D
	variable  pbDFactor =1		// value of 1 means Ca buffer diffuses as fast as free Ca
	nvar dx, pdt
	variable cnt
	variable t =0
	variable /g now =datetime
	variable BuffOn, Buffoff, PumpOut, extruded
	variable /g  BuffFactor
	BuffOff = 3.3 * pdt			//reciprocal of this rate constant is residence time in sec
	BuffOn = BuffOff*BuffFactor
	variable pdtDdx= pdt*D/dx, pdtbDdx = pdtDdx * pbDFactor
	PumpOut = 127*pdt		// sink at origin = pumping, units 1.e-21 mol/s
	do					//step through time
		cnt = 0
		do					//calculate fluxes
			pFlux[cnt] = -pdtDdx * (pcCa[cnt] - pcCa[cnt+1])
			pBFlux[cnt] = -pdtbDdx * (pcCaBuff[cnt] - pcCaBuff[cnt+1])
			ToPBuff[cnt] = BuffOn * pMCa[cnt] - BuffOff * pMCaBuff[cnt]
			Cnt += 1
		while (cnt < plim)

		pMCa[0] += -PumpOut		//make a sink at origin = pumping, units 1.e-21 mol/s

		cnt = 0
		do			//increment mass and concentrations
			pMCa[cnt] += pFlux[cnt] - ToPBuff[cnt]
			pMCa[cnt+1] += -pFlux[cnt]
			pCCa[cnt] = pMCa[cnt] / dx
```

```
                pMCaBuff[cnt] += pBFlux[cnt] +   ToPBuff[cnt]
                pMCaBuff[cnt+1] += -pBFlux[cnt]
                pCCaBuff[cnt] = pMCaBuff[cnt] / dx

                Cnt += 1
            while (cnt < plim)

            t +=pdt

        while (t <100000*pdt)   // insert number of time iterations here

        extruded = PumpOut*t/pdt
        variable /g now
        print datetime - now, "seconds;", "  extruded", extruded//   timing of calculations
        variable temp1, temp2, Mass = 0  //----solve analytical diffusion equation for point source
        cnt = 0
        Mass = sum (pmca)               //total; of free calcium
        cnt = 0
        do                              //increment distance in evaluating analytic solution
            Temp1 = exp( (-(dist[cnt])^2)/(4*D*t))
            Temp2 = (4*pi*D*t)^0.5
            pTheory[cnt] = 2* Mass * Temp1/Temp2//factor of 2 for hemisphere
            cnt += 1
        while (cnt < plim)

        Mass = sum (pmca) +  sum (pmcabuff)                 //total; of free and bound calcium
        variable /g StartMass
        print StartMass, "dMass", StartMass-Mass,"t", t-pdt, "[Ca]" , pcCa[0], pcCa[plim-1],"D", D,
D*pbDFactor,"K", BuffFactor, buffoff/pdt
end
```