WILEY-VCH

ADVANCED HEALTHCARE MATERIALS

Supporting Information

for *Adv. Healthcare Mater.*, DOI: 10.1002/adhm.202101826

Wireless Force-Inducing Neuronal Stimulation Mediated by High Magnetic Moment Microdiscs

Claudia Collier, Nicolas Muzzio, Rohini Guntnur, Amanda Gomez, Carolina Redondo, Raquel Zurbano, Ivan K. Schuller, Carlos Monton, Rafael Morales and Gabriela Romero**

Supporting Information

Wireless Force-Inducing Neuronal Stimulation Mediated by High Magnetic Moment Microdiscs

Claudia Collier, Nicolas Muzzio, Rohini Guntnur, Amanda Gomez, Carolina Redondo, Raquel Zurbano, Ivan K. Schuller, Carlos Monton, Rafael Morales and Gabriela Romero**

The following videos are available:

- **Video S1**. MMDs behavior under applied AMFs.
- **Video S2**. Force-inducing hippocampal neural stimulation mediated by MMDs.
- **Video S3**. Force-inducing cortical neural stimulation mediated by MMDs.
- **Video S4**. Force-inducing cortical neural stimulation mediated by MMDs on neurons pretreated with the global inhibitor for mechanosensitive channels Gd^{3+}.

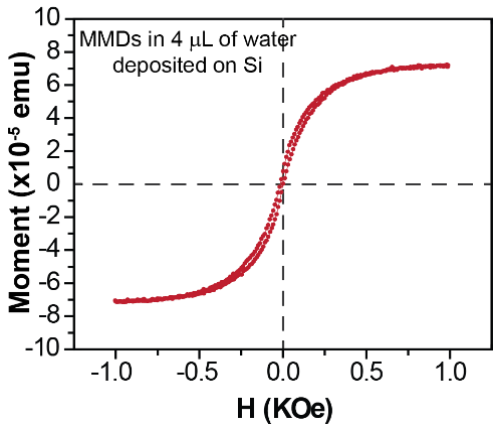

Figure S1. Hysteresis loop of released MMDs dispersed and dried on a Si substrate (estimated concentration ~ 6.4 x 10^4 MMDs mL^{-1}).

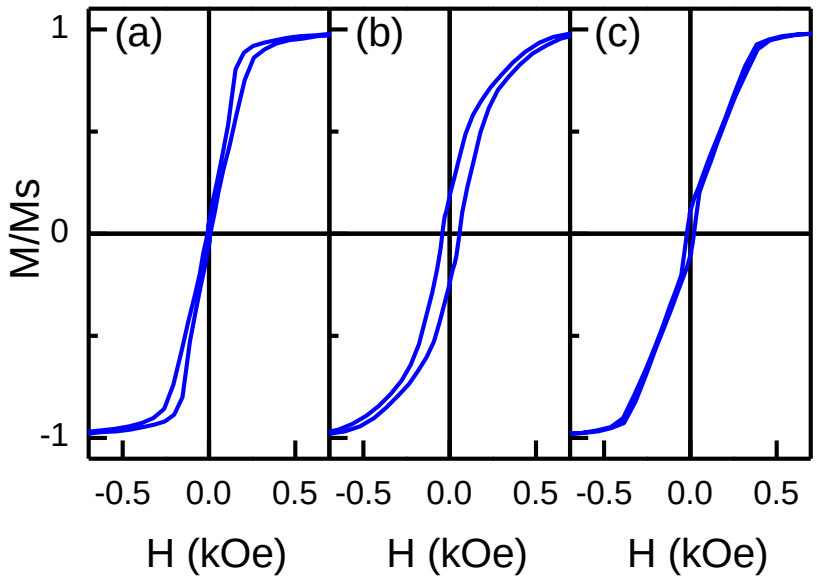

Figure S2. Hysteresis loops of Permalloy ($Ni_{80}Fe_{20}$) MMDs of diameter 4 μm and thickness; (a) Py 100 nm; (b) Py 200 nm; (c) stacking structure Py (100)/Ti (20)/Py (100) [nm].

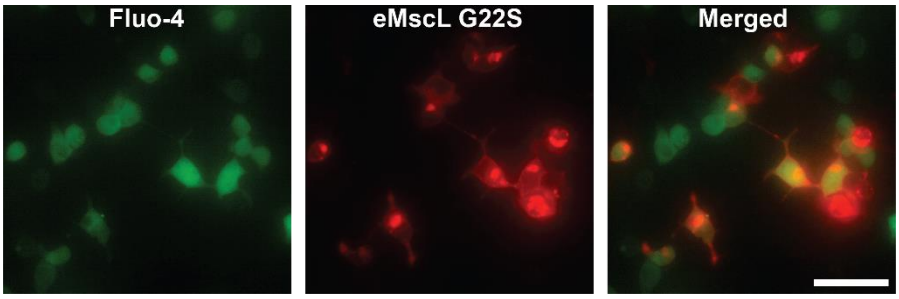

Figure S3. HEK293 cells labelled with the calcium indicator Fluo-4 (green) and transfected with the large conductance mechanosensitive ion channel eMscL G22S (red) using Lipofectamine 3000. Merged image shows cells label with both Fluo-4 and eMscL. Particularly, eMscL is shown to be inserted only in the cell membrane. Scale bar = 50 μm.

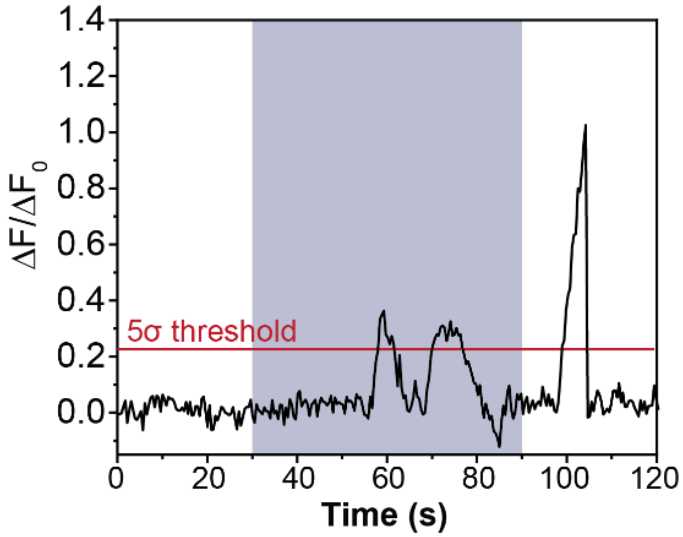

Figure S4. The Calcium fluorescence intensity for a representative neuron recorded during a 120 second trial. A threshold set at five standard deviations (5σ) above background noise (red line) was used to classify neurons as momentarily active or inactive.

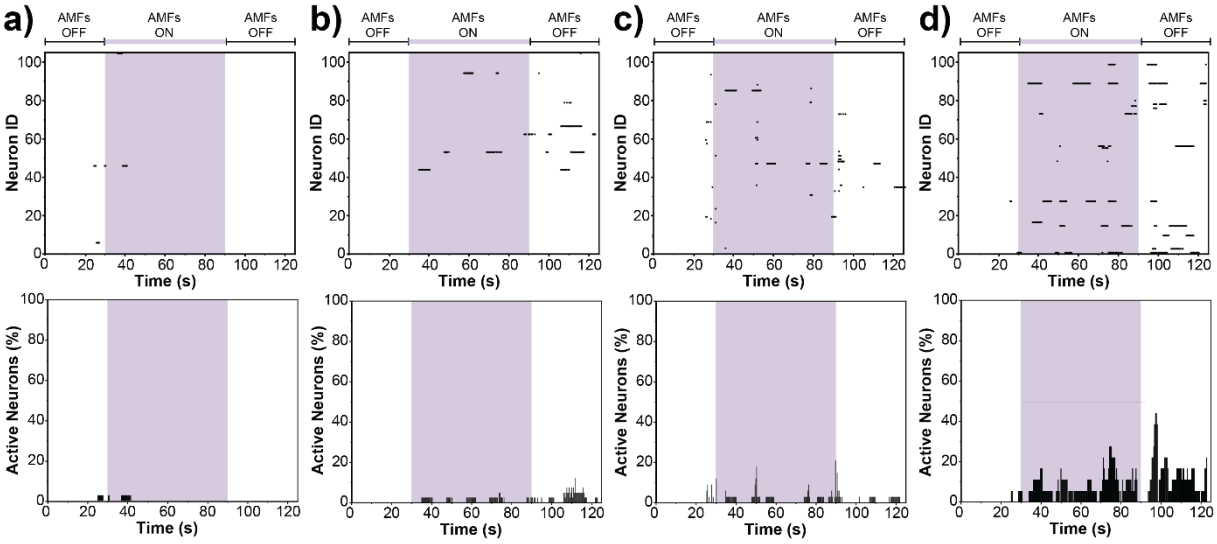

Figure S5. Raster plot tracking and percentage of active neurons from 100 random hippocampal neurons: (**a**) expressing eMscL in the absence of MMDs and AMFs; (**b**) expressing eMscL in the absence of MMDs and subjected to AMFs stimulation; (**c**) expressing eMscL, co-cultured with MMDs and with no AMFs stimulation; and (**d**) not transfected with eMscL in the presence of MMDs and AMFs.

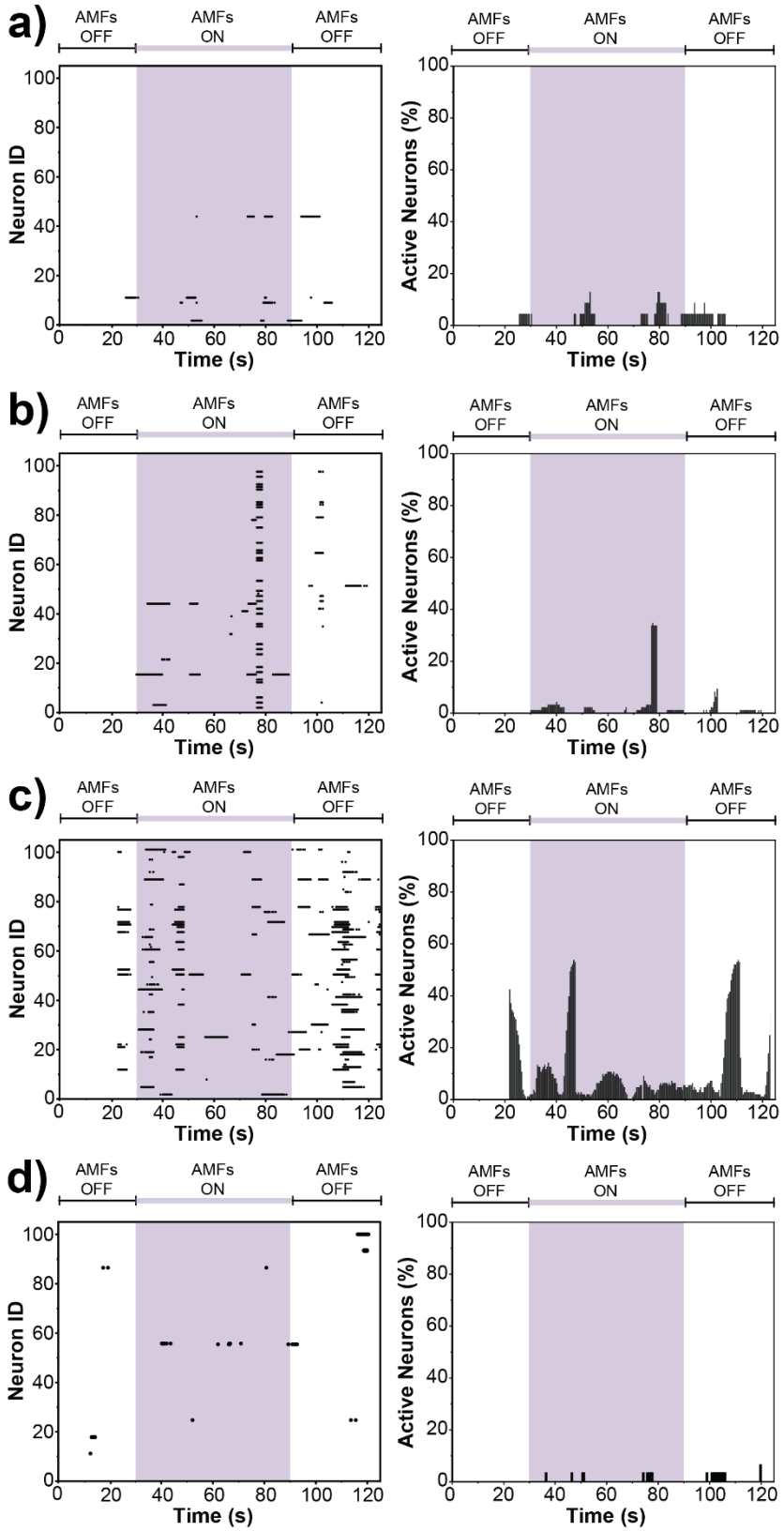

Figure S6. Raster plot tracking and percentage of active neurons from 100 random cortical neurons: (**a**) in the absence of MMDs and AMFs; (**b**) co-cultured with MMDs and with no AMFs stimulation; (**c**) in the absence of MMDs and subjected to AMFs stimulation; and (**d**) pretreated with Gd^{3+}, in the presence of MMDs and subjected to AMFs stimulation.

Supplementary Note. Algorithm for quantifying neural activity based on the Calcium fluorescence intensity of Fluo-4.

The intensity vs time data was acquired by the 'Time Series Analyzer' Image J plugin for each cell in each video. These data were then analyzed using a MATLAB code executing the following algorithm for each cell:

1. The data was divided into 5 equal segments and each segment was fitted into a polynomial regression to subtract the baseline.
2. The first 20 s of the data set were used as a reference for background noise and a threshold of 5 times standard deviation (5σ) was chosen to identify neural activity. See Figure S3 for an example of fluorescence thresholding.
3. All values above the 5σ threshold were considered as neural activity and a matrix of 0 (no activity) and 1 (activity) was generated.
4. Finally, because different numbers of cells were counted in each video across 5 trials, a sample containing the activity information of 100 cells was automatically randomly picked.

The MATLAB code is provided below:

```
function output = checkDataSample(time, noisy_data, num_sample)
% CHECKDATASAMPLE Detrends a random sample of noisy data and returns a boolean
% matrix such that output(i,j) == 1 if the detrended value for column j time step i > 5 standard
% deviations of the first 20 seconds.
	clean_data = noisy_data;
	data_size = size(noisy_data);
	for i=1:data_size(2)
		start = 0;
		batch_size = size(noisy_data,1)/5;
		f_y = noisy_data(:,i);
		for j=0:4
			start = j*batch_size+1;
			last = j*batch_size+batch_size;
			% if last is > size of data, then make last = last row of noisy data.
			if last > size(noisy_data, 1)
				last = size(noisy_data,1);
			end
			size(noisy_data(start:last, i));
			% Fits a linear regression for column i from time start to last.
			[p,s,mu] = polyfit(time(start:last),noisy_data(start:last,i),1);
			f_y(start:last) = polyval(p,time(start:last),[],mu);
		end
		% Subtracts the linear trends from the noisy data.
		clean_data(:,i) = noisy_data(:,i) - f_y;
		% Plots the noisy and detrended data.
		plot(time, clean_data(:,i));
		figure;
		plot(time, noisy_data(:,i));
	end

	output = clean_data;
	% Calculates the standard deviations for the first 20 seconds.
```

```
        std_devs = std(clean_data(time<=20,:));
        % Creates the boolean output matrix (1 if output(i,j)> 5*std_dev otherwise 0).
        for i=1:length(std_devs)
                output(:, i) = output(:, i) >= 5*std_devs(i);
        end

        numCols = size(output,2);
        idxs = 1:numCols;
        % Samples without replacement num_sample columns of the output matrix.
        sampleIdxs = datasample(idxs, num_sample, 'Replace', false);
        size(output)
        output = output(:,sampleIdxs);
end
```