Decoding context

In the file `DecodingDataRead.m`, we first took the binarized calcium activity and took the sum of events in 3 s bins (60 frames). We then restricted analysis to:

(1) times when the mouse was running, and

(2) when the mouse was in context "1" or "2," representing two different cue configurations in the maze.

Finally, only activity from place cells were used. The resulting pre-processed data was saved for each mouse in a subfolder under the name `binned_activity_decode.mat`.

```
rootdir = "";
folders = ["M119"+filesep "M120"+filesep "M292"+filesep "M319"+filesep ...
    "M231"+filesep "M314"+filesep "M316"+filesep "M318"+filesep "M210"+filesep];
cohort = ["D1","D1","D1","D1","G1","G1","G1","G1","D1"];
sessions = ["early", "trained", "grouping"];
```

Estimate SVM performance for each animal & each condition.

We saved each run as a separate .mat file manually for comparision later. The resulting `validation_accuracy` scores were put into excel tables for glme analysis and ease of comparision.

For panel D, the `ablate_percent` and `ablation` strategy were varied.

```
ablate_percent=0.50;
ablation = 'descend';

test_accuracy = nan(length(folders),3);
validation_accuracy = nan(length(folders),3);
s_max = length(sessions);

weights = cell(length(folders),3);
corrs = cell(length(folders),3);

parfor f = 1:length(folders)
    for s = 1:s_max
       try
            A=load(folders(f)+"binned_activity_decode_"+sessions(s)+".mat");

            ds = 10; % how many samples to downsample

            N_cells = size(A.a_use,2);
            N_cells_use = round(ablate_percent*N_cells);
            idx_cells_use = [];
            if strcmp(ablation,'random')
                % shuffle the cells
                subset_cells = randperm(N_cells);
                idx_cells_use = subset_cells(1:N_cells_use);
                corrs{f,s} = A.corr_coef(idx_cells_use);
```

```
            elseif strcmp(ablation, 'ascend')
                % sort the cells by increasing corr coef (most remap are
                % first)
                [corr_temp,idx_cells_use] =
topkrows(A.corr_coef,N_cells_use,'ascend');
                corrs{f,s} = corr_temp;

            elseif strcmp(ablation,'descend')
                [corr_temp,idx_cells_use] =
topkrows(A.corr_coef,N_cells_use,'descend');
                corrs{f,s} = corr_temp;
            elseif strcmp(ablation,'None')
                idx_cells_use = 1:N_cells_use;
                corrs{f,s} = A.corr_coef(idx_cells_use);
            end

            a_use_sub = A.a_use(1:ds:end,idx_cells_use);
            context_use_sub = A.context_use(1:ds:end);

            % % reshuffle context
            % shuffle_context_ind = randperm(size(context_use_sub,1));
            % context_use_sub = context_use_sub(shuffle_context_ind);

            hpartition =
cvpartition(context_use_sub,'Holdout',0.25,'Stratify',true);

            X_train = a_use_sub(training(hpartition),:);
            Y_trian = context_use_sub(training(hpartition));

            classificationSVM = fitcsvm(...
                X_train, ...
                Y_trian, ...
                'KernelFunction', 'rbf', ...
                'PolynomialOrder', [], ...
                'KernelScale', 'auto', ...
                'BoxConstraint', 1, ... % usuaally 1
                'Standardize', true);
                %'ScoreTransform','sign');

            % k-fold partitioning validation
            partitionedModel = crossval(classificationSVM, 'KFold', 20);
            [validationPredictions, validationScores] =
kfoldPredict(partitionedModel);
            validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun',
'ClassifError');

            validation_accuracy(f,s) = validationAccuracy;

            % test set on hold out
            X_test = a_use_sub(test(hpartition),:);
```

```
            Y_test = context_use_sub(test(hpartition));

            [testPredictions,testScores] = predict(classificationSVM,X_test);

            testAccuracy = 1 - loss(classificationSVM,X_test,Y_test);

            rocObj_test = rocmetrics(Y_test, testScores,[1,2]);

            test_accuracy(f,s) = testAccuracy;

            % comparing SVM weights to remapping metric
            weights{f,s} = classificationSVM.Beta;

            display("Completed "+ folders(f)+"binned_activity_decode_"+sessions(s)
+".mat")

        catch e
            display("Skipping "+ folders(f)+"binned_activity_decode_"+sessions(s)
+".mat")
        end
    end
end
```

```
"Completed M292\binned_activity_decode_early.mat"

"Completed M120\binned_activity_decode_early.mat"

"Completed M319\binned_activity_decode_early.mat"

"Completed M316\binned_activity_decode_early.mat"

"Completed M119\binned_activity_decode_early.mat"

"Completed M231\binned_activity_decode_early.mat"

"Completed M292\binned_activity_decode_trained.mat"

"Completed M292\binned_activity_decode_grouping.mat"

"Completed M316\binned_activity_decode_trained.mat"

"Completed M119\binned_activity_decode_trained.mat"

"Completed M319\binned_activity_decode_trained.mat"

"Completed M120\binned_activity_decode_trained.mat"

"Completed M314\binned_activity_decode_early.mat"

"Completed M316\binned_activity_decode_grouping.mat"

"Completed M318\binned_activity_decode_early.mat"

"Completed M119\binned_activity_decode_grouping.mat"

"Skipping M210\binned_activity_decode_early.mat"

"Completed M231\binned_activity_decode_trained.mat"
```

```
"Completed M120\binned_activity_decode_grouping.mat"

"Completed M319\binned_activity_decode_grouping.mat"

"Completed M231\binned_activity_decode_grouping.mat"

"Completed M210\binned_activity_decode_trained.mat"

"Skipping M210\binned_activity_decode_grouping.mat"

"Completed M314\binned_activity_decode_trained.mat"

"Completed M318\binned_activity_decode_trained.mat"

"Completed M318\binned_activity_decode_grouping.mat"

"Completed M314\binned_activity_decode_grouping.mat"
```

Stats for Panel B

```
clear;
datat = readtable("decoder_results_deltas_repeat.xlsx");
WithinSubjectData =
table(categorical({'All','PC','NonPC','All','PC','NonPC'})',categorical({'D','D','D'
,'G','G','G'})',VariableNames=["CellSets","Tasks"]);

rm = fitrm(datat,'Score1-Score6 ~ 1','WithinDesign',WithinSubjectData);
ranova(rm,'WithinModel','CellSets+Tasks')
```

ans = 6×8 table

...

	SumSq	DF	MeanSq	F	pValue	pValueGG
1 (Intercept)	35.2169	1	35.2169	2.2641e+03	4.7363e-10	4.7363e-10
2 Error	0.1089	7	0.0156	1	0.5000	0.5000
3 (Intercept):CellSets	0.0907	2	0.0453	11.0551	0.0013	0.0084
4 Error(CellSets)	0.0574	14	0.0041	1	0.5000	0.5000
5 (Intercept):Tasks	0.0060	1	0.0060	8.9293	0.0203	0.0203
6 Error(Tasks)	0.0047	7	6.7139e-04	1	0.5000	0.5000

```
rm.multcompare('Tasks','by','CellSets')
```

ans = 6×8 table

...

	CellSets	Tasks_1	Tasks_2	Difference	StdErr	pValue	Lower
1	All	D	G	0.0165	0.0064	0.0361	0.0014
2	All	G	D	-0.0165	0.0064	0.0361	-0.0316
3	NonPC	D	G	0.0162	0.0107	0.1729	-0.0090
4	NonPC	G	D	-0.0162	0.0107	0.1729	-0.0414

	CellSets	Tasks_1	Tasks_2	Difference	StdErr	pValue	Lower
5	PC	D	G	0.0343	0.0106	0.0143	0.0092
6	PC	G	D	-0.0343	0.0106	0.0143	-0.0594

```
rm.multcompare('CellSets','by','Tasks')
```

ans = 12×8 table

...

	Tasks	CellSets_1	CellSets_2	Difference	StdErr	pValue	Lower
1	D	All	NonPC	0.1045	0.0223	0.0055	0.0389
2	D	All	PC	0.0251	0.0088	0.0564	-7.8038e-04
3	D	NonPC	All	-0.1045	0.0223	0.0055	-0.1701
4	D	NonPC	PC	-0.0794	0.0278	0.0561	-0.1612
5	D	PC	All	-0.0251	0.0088	0.0564	-0.0510
6	D	PC	NonPC	0.0794	0.0278	0.0561	-0.0023
7	G	All	NonPC	0.1042	0.0243	0.0089	0.0326
8	G	All	PC	0.0429	0.0150	0.0565	-0.0013
9	G	NonPC	All	-0.1042	0.0243	0.0089	-0.1758
10	G	NonPC	PC	-0.0613	0.0326	0.2141	-0.1573
11	G	PC	All	-0.0429	0.0150	0.0565	-0.0871
12	G	PC	NonPC	0.0613	0.0326	0.2141	-0.0347

Stats for Panel D, inset

The reported beta values are the interaction terms between the percentage "slope" and each task type.

```
clear;
datat = readtable("decoder_results_deltas_repeat_ablate.xlsx");
percents = [zeros(9,1);repmat(25,9,1);repmat(50,9,1)]/100;
cohorts = repmat(datat.Cohort,3,1);
animals = repmat(datat.Animal,3,1);
DU_scores = [datat.Score1;datat.Score5;datat.Score9];
DS_scores = [datat.Score3;datat.Score7;datat.Score11];
GU_scores = [datat.Score2;datat.Score6;datat.Score10];
GS_scores = [datat.Score4;datat.Score8;datat.Score12];

grand_tasks =
reordercats(categorical([repmat("DU",27,1);repmat("DS",27,1);repmat("GU",27,1);repma
t("GS",27,1)]),["DU","DS","GU","GS"]);
grand_scores = [DU_scores;DS_scores;GU_scores;GS_scores];
grand_percents = repmat(percents,4,1);
grand_animals = repmat(animals,4,1);
grand_cohorts = repmat(cohorts,4,1);
```

```
grand_table =
table(grand_animals,grand_cohorts,grand_percents,grand_tasks,grand_scores);

grand_glme = fitglme(grand_table,'grand_scores ~ 1 + grand_percents*grand_tasks +
(1|grand_animals)')
```

```
grand_glme =
Generalized linear mixed-effects model fit by PL

Model information:
    Number of observations             102
    Fixed effects coefficients           8
    Random effects coefficients          9
    Covariance parameters                2
    Distribution                    Normal
    Link                            Identity
    FitMethod                       MPL

Formula:
    grand_scores ~ 1 + grand_percents*grand_tasks + (1 | grand_animals)

Model fit statistics:
    AIC       BIC        LogLikelihood    Deviance
    -417.7    -391.45    218.85           -437.7

Fixed effects coefficients (95% CIs):
    Name                                    Estimate     SE          tStat       DF    pValue         Lower        U
    {'(Intercept)'                  }         0.89465    0.024998      35.789    94    1.4748e-56      0.84501
    {'grand_percents'               }        -0.11178    0.021694     -5.1524    94    1.4149e-06     -0.15485
    {'grand_tasks_DS'               }       0.0013544    0.009902     0.13678    94       0.89149    -0.018306
    {'grand_tasks_GU'               }       -0.026374    0.010266     -2.5692    94      0.011767    -0.046757    -
    {'grand_tasks_GS'               }       -0.028085    0.010266     -2.7358    94     0.0074402    -0.048467    -
    {'grand_percents:grand_tasks_DS'}       -0.012929     0.03068    -0.42141    94       0.67442    -0.073846
    {'grand_percents:grand_tasks_GU'}       -0.017113    0.031625    -0.54112    94       0.58971    -0.079904
    {'grand_percents:grand_tasks_GS'}        -0.08602    0.031625     -2.7201    94     0.0077758     -0.14881

Random effects covariance parameters:
Group: grand_animals (9 Levels)
    Name1                 Name2                 Type          Estimate
    {'(Intercept)'}       {'(Intercept)'}       {'std'}       0.071993

Group: Error
    Name                        Estimate
    {'sqrt(Dispersion)'}        0.02301
```