**Multivariate Analysis**

**Ajay Attuchirayil Krishnankutty**

**December 2018**

## INTRODUCTION

This report consists of details about various multivariate analytical techniques being applied on 3 different datasets.

The techniques used here are as follows:

* One-Table
  + PCA (Principal Component Analysis)
  + CA (Correspondence Analysis)
  + MCA (Multiple Correspondence Analysis)
* Two-Table
  + PLS-C (Partial Least Square Correlation)
  + BADA (Barycentric Discriminant Analysis)
  + DiCA (Discriminant Correspondence Analysis)
* Multi-Table
  + DiSTATIS
  + MFA/STATIS (Multiple Factor Analysis)

Three datasets were used for working with above mentioned techniques, which are as follows:

* IBM-HR-EMPLOYEE-WITH-ATTRITION
* BIRD\_BRAIN
* 19\_EXPERT-18\_WINES-3\_COLOR

The IBM data set has been used to do analysis using PCA, MCA, PLS-C, BADA, DiCA, MFA

Bird\_Brain was used to do CA

And, last dataset was used to analysis using DiSTATIS.

Each of the data set has been explained briefly, followed by the techniques applied on it, in the same order as mentioned above.

## Data set: IBM-HR-Employee-With Attrition

The dataset given is regarding IBM employees, and this data consists of details of IBM employees who have faced attrition (relieving and employee of his/her duties).

Number of employees in the dataset (**ROWS**)**: 237**

Number of variables measured for these employees (**COLUMNS**): **32**

The variables/columns have a mixture of quantitative and other types as follows:

* **Quantitative:** Age, Monthly Income, Daily Rate, Hourly Rate, Distance from Home, Num of Companies Worked, Percent Salary Hike
* **Ordinal:** Performance Rating, Job Satisfaction, Relationship Satisfaction, Job Level
* **Nominal:** Marital Status, Education Field, Department, Job Role, Gender, Overtime

This dataset has details of employees who have faced attrition, hence the attrition column has value ‘Yes’ for all rows.

Running the *head* command on few selected columns gives us the following result:

![A screenshot of a cell phone

Description generated with very high confidence](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAkACQAAD/4RDcRXhpZgAATU0AKgAAAAgABAE7AAIAAAAGAAAISodpAAQAAAABAAAIUJydAAEAAAAMAAAQyOocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGFqYXljAAAFkAMAAgAAABQAABCekAQAAgAAABQAABCykpEAAgAAAAMwMwAAkpIAAgAAAAMwMwAA6hwABwAACAwAAAiSAAAAABzqAAAACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMjAxODoxMTowNyAxNDowNDo0OAAyMDE4OjExOjA3IDE0OjA0OjQ4AAAAYQBqAGEAeQBjAAAA/+ELGGh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8APD94cGFja2V0IGJlZ2luPSfvu78nIGlkPSdXNU0wTXBDZWhpSHpyZVN6TlRjemtjOWQnPz4NCjx4OnhtcG1ldGEgeG1sbnM6eD0iYWRvYmU6bnM6bWV0YS8iPjxyZGY6UkRGIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iLz48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOnhtcD0iaHR0cDovL25zLmFkb2JlLmNvbS94YXAvMS4wLyI+PHhtcDpDcmVhdGVEYXRlPjIwMTgtMTEtMDdUMTQ6MDQ6NDguMDMxPC94bXA6Q3JlYXRlRGF0ZT48L3JkZjpEZXNjcmlwdGlvbj48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOmRjPSJodHRwOi8vcHVybC5vcmcvZGMvZWxlbWVudHMvMS4xLyI+PGRjOmNyZWF0b3I+PHJkZjpTZXEgeG1sbnM6cmRmPSJodHRwOi8vd3d3LnczLm9yZy8xOTk5LzAyLzIyLXJkZi1zeW50YXgtbnMjIj48cmRmOmxpPmFqYXljPC9yZGY6bGk+PC9yZGY6U2VxPg0KCQkJPC9kYzpjcmVhdG9yPjwvcmRmOkRlc2NyaXB0aW9uPjwvcmRmOlJERj48L3g6eG1wbWV0YT4NCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgPD94cGFja2V0IGVuZD0ndyc/Pv/bAEMABwUFBgUEBwYFBggHBwgKEQsKCQkKFQ8QDBEYFRoZGBUYFxseJyEbHSUdFxgiLiIlKCkrLCsaIC8zLyoyJyorKv/bAEMBBwgICgkKFAsLFCocGBwqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKv/AABEIALMDkAMBIgACEQEDEQH/xAAfAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJCgv/xAC1EAACAQMDAgQDBQUEBAAAAX0BAgMABBEFEiExQQYTUWEHInEUMoGRoQgjQrHBFVLR8CQzYnKCCQoWFxgZGiUmJygpKjQ1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6wsPExcbHyMnK0tPU1dbX2Nna4eLj5OXm5+jp6vHy8/T19vf4+fr/xAAfAQADAQEBAQEBAQEBAAAAAAAAAQIDBAUGBwgJCgv/xAC1EQACAQIEBAMEBwUEBAABAncAAQIDEQQFITEGEkFRB2FxEyIygQgUQpGhscEJIzNS8BVictEKFiQ04SXxFxgZGiYnKCkqNTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqCg4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2dri4+Tl5ufo6ery8/T19vf4+fr/2gAMAwEAAhEDEQA/APon7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/CrFFAFf7GP+esn6f4UfYx/z1k/T/Co552a6EKHb2yG61NBvVnR23YAI/HP+FADfsY/56yfp/hR9jH/PWT9P8KsVT1K5ktrcGMrGGOGmYbhH9FHLE9AB/wDrAJPsY/56yfp/hR9jH/PWT9P8KLKd7m1WSVVBPRkbKuP7y+x9+asUAV/sY/56yfp/hR9jH/PWT9P8Kp6zrP8AYojmni8y3fK/KwDh8ZHB7YBye3U8ZIv2zSvaxtcKiysoLCNtyg+x70AM+xj/AJ6yfp/hR9jH/PWT9P8ACrFFABRRRQAUVzXj3xEPDfhWW4S+ttPubmRbW2ubplEcMjnAdt3GFGWOeuMVznhP4ko/wx1HWNZu49WvNDlktrqWyKMLt1bCOmz5BvBU8cDPtQB6RRXFQ+Pbyy1KWz8T6NDpxh0l9Uke2vjcAIrY2cxp836ehPWovBfxNtPF2s/2aEsFlktBeQ/YdSW72pnBSXCr5cgyPl+YcnnigDuqKK4vU9U1XWfiM3hbS9Tk0e3stPW+urm3iieaUu7KiJ5isqqNpJO0k8AEdwDtKK4fV/HNl4R0C1a41uz1qa5vmsYbq5uYraNXAJYTyoCiFdpyQg5wNuaqQfFVbzQNLvtP0qO8nvtWOlNHBfK0SyDPzpKFw6cAg4BwemeKAPQ6K4NPiTKvh69uJ9GX+1rXVRpK2EN3vjlnLAKRKUUhcHJOzIAPBrHi8bXvh7xL401DxQGji061sytlFeGWFHcEfIzBQASRklV9xxQB6pRXm6fF63PhXxHqa2lheXOgwpM8em6mLm3mVxxtnCDB6ggrxjvmu70m6vL3T0uNQtIrR5PmSKOcy4QjI3HauG9QMgepoAu0UUUAFFFcRrniq70f4o2ljPdFNHXQrq/uIRGpJeN0AbOM8AnjOKAO3orh9I8fXt7daQmr6GunW2vQPLps0V55zfKu8LKuxfLYpz8pccEZ6ZwfAnjfV7Pwn4dfXLKS7stSvHsv7RkvjJOJTI+wtGVPyfKBnfkf3cUAerUV4va+NPEl34ylsLLXLyW9/wCEiktItOmsYks2s0wZP33lAl1UkgLIWzjK4rsNW+IlxZ3munTdFW+0/wAOlRqc7XRjkBKh2EUexhIVQgnLJzwKAO4orze98YJpXxB1PU7q8uW0W28MpqHkKx2/6wncEJxuIwPWtLwV8R7bxdq9xpuywE8Vql2j6dqS3sexiRtdgq7JAcZXBHPBNAHbUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRXkmi/EHXLLVdeg166ju1uZLj+wlaNI/njnaHyMqBuOfLbJycE+lT+EfiBrMPgewOuMuteIL/AFm40u3Hy20cjI7jLFUO1QqE5Ck9OtAHqlFef3/xPl0zRr6W70MnVNO1O3065so7rKEzMux45Sg3DDA8qvOQcdabqfxD1a28MeMZ49ItLfV/DSbjE9000EqmMOG3BFY8Z+XA5GN3egD0KivMdQ8ReJJPFPgiSK3j+06hYXMs1hDfulq7eWrKXYrnAz/cYgnv1qO78Z3XiCX4e6jZmfTVv9Xlhu7WO4JVtkUoKMRjeu5cjI9DigD1KiuP0rx6dUtYkXTPK1NtVk02Sya4z5ZjJLybtvTyxuHHOQO+a5/Q/iJqtv4Qtr7ULc6vd3niKfSowHSDYnnOqHhcHaFHufXNAHqFFcGnxOWDQdWudT0sRajpmprpRs7a5MqTXD7fLCyFF4O8ZJXjnrV/VPF2p6Nb6Za32i239t6rdm1tLWK/LQMQpcuZjEGChQc/u85GBkc0AdbRXn978UHsNHuJJ9E3apZ6xBpN1ZJdfIryldrpKUG5SrAjKr6HFV0+Kl7HPOmoeHoYEs9bj0e7kj1AybXk27HjHlAuPmGQduOxPOAD0iivPPFHxc07w34hu9NZLF10/wAr7WJ9RWCdvMxxBCVJlIByeV9BmvQkcSRq68qwBFAC0UUUAFFFFABRRRQAUUVn6zYXV/ZhbLWL3SnQ7jLZpCzOMH5T5sbjHfgA8daANCivL/CvibUdP+GR8beJ9e1LVdsUu6x8m1RGYSlE27IlbccActjk8Vst431iBtQ0/UdDtbLV4tLbUrNFvmmt5lXgqz+WrKynGRtIOeD6AHb0VwGnePdVPwx0bxNq0Hh+0nv4kkf7Xq7WduAy5GHaJjuP9znHPzGs+58Y3HiHUPhtqenyXFjbateT/aLaOclXAgf5WIwHUMMjI9DgUAen0V5DcfEPxDYQ2dvK6TXOmaxLHrcjIq/6GJVRGxjAJE0bcY4Rvx6BPHF7bQxSrEuo3GuavLaaNbM4hjEUYILNIEY7T5bvnDHkAcUAd9RXn978UHsNHuJJ9E3apZ6xBpN1ZJdfIryldrpKUG5SrAjKr6HFZfifxpdeGPifoza1I0Kz6LOF0u1uy8dzdGZFiRdwQM5zgEqMZPYE0AeqUVxeo6tcxeOvCttfx3Ntc3VpcyS29rfk26sqKSGUoPNxnhvlx1xVHQfifc6tdeHjeaCllZ69LPBbyi98yRZItxOU8sDaQhwd2fUUAehUV57cfE66W1bU7TQo59GOrppMVy98UldzMImfyvLICBt2PmycdBnNQav4j8U6B4vu/DgkbUJddHmaBdyQKEtCBiVJdoAIQfOCeWHy5zQB6TRXnXir4p2vgzUJNImezvbrT7JLm8e+1COzknByMQpsIkkIUnb8gHHPNWvFPxLj8NSWczRaVLZ3SQyJHLqohvZFkYDdHbmM7wMj+Md/SgDu6K5ZfFt5cfEK78MWOkxyLaWsF1Ldy3ewBJGYEBAhJYbeBnB5yV70/CfxDXxT4o1PSVtrOybT5pYWt571hfHYQN5tzGAEOc7g7dR3OAAdrRRRQAUUUUAFFFFABRRRQAUUVyHi/XdQ0rxj4NsrK58q11K+miu02KfMRYGcDJBI5APGKAOvorgrP4lT3IsdRl0QReH9Q1A2Fvei73Tb9xRXeHYAEZlIBDseQSOuMTR/G2s6HbeJNRvrKTUtKs/EEsE08t8fNgiLIo8qMqQyruyQWTHYGgD1iivNLvxbqOlfEx7PxPqd7o2m3E8UejstnE9heKVGUklKl1lJ3YG9AML16HV1T4hf2R43s9BuotKkW8ultkWDVg14m5ch3tjGMLkdd54IPtQB2H2WMSGRQVYnJPX+fSpEj2EnJYnjJrzafxnrV7H8QrXULSC3stBRkhn0+8eO5OYQ4+YoQDg53D7p42t1qe4+I66LN4a0loIEbU7BJ1vdZ1FoI24UbBKImDy85wduePXFAHolMeGOSRJHQM0ZJQn+EnjNed31zrmo/HKLTUdl02y01blUh1OWAZaTBdo1QrKeMBHO3HOQeKu6N8Sor/xlN4fvE0ovHaS3Rm0zVReCIRsFZJh5aGNvmBx83Q+lAHbxQxwhhEgQMxYgdMnqafXGad4z1fWvCV34h07RbaOwe0kudNknvW3zBQdplQR/uwwGRtZz64rCtPF+o6xpnw51LWYpLS51q73GPS75khYGBmHmIyEsv+xng4O44oA9La2he4E7xK0qoUDEZIU9R+OKdBBHbQJDAgjjQYVV6AelcInxOZoxqv8AZC/8I42p/wBmC+F0TP5nmeVv8nZjZ5ny53574qC++KdxZXWqs2gI2n6TrMWlXFx9uw58zYA6R+XzgyDILDjoTyAAei0UUUAFFFIw3KQc4IxwcUAZl5YWNz4gsLu5uiLiySV4bbzFCncApkK4ySBkA5wNx9a5bV9A0TU5PEd7b77+LWLSC3uIIbtVhlYMyrIrBTtkHTdk9BxxXTnRWit7qC1nURXKncJY9z7sAAlwQWGBzu3MfXtUUmmXtylxJPDZxyymEiJXZ0+Rs8kqM5HtQBw3hjR7m/8AFbapqqX1zpc+mtpYbVbu2mlmGS7fLakxKoC7c5DEk56ZrtPDPh6PRreL7Jr2o6nYpFstIbmaJ44U7BWRFZ+AAC7OcDr1zag066+0i6m8mOVpjK8aMWVf3ewYOBnsegqxZaYLa4kupZd9xLzIY18uMnAGdoPPTqxYjsccUAXq5/xB4asb6+h1v+0rrRb+0jMf9oWkiKfKPJRxIrRsuefmU4PQiugqheaYLi5S6hl8u5i5jMi+agOCPunp16qVJ7kjigDjrbwpoK2UujQXjQjTbptSbVvtiC6iunJZptuzywDuIPAXkjbitK50bR75tEh1PxDc311b3Y1C2nlmi3XDKNo4VAm3B6IFzjPXJram024f7W6NEXkmjmjDZ2koF4b0yR74pXtLx7i2uPLt1cJIkyLIcLvwcg7fmOR3AzmgDn5fBPh29tdQsk1GfzdR1EamssVwnmwXC7WDR/LjgAHDBuDzkGpo/hzpTnWW1K81DU5Naiiiu5LmVQT5YwrJ5aqEbv8ALgAjIA5rYTRtuk22ngwxQIgEqrCGLMMHI3cDnnJBP0PNaUMSwQrGhcqowC7lj+JJJNAHNXngePVPCuo6DrWvavqUF/CIHmneFZI0HZdkarn/AGmUk+vSulhiEEEcSZKxqFBPXAGKfRQAUUUUAFcbceHJrzx9Jr3iM6fDYJYyaZaxJOzNOsrgkvuVQpIAAUFu/PauyqhJpmNRF7bTbJvut5yeaNpIyFydy9OgIXuVNAHG+HfDei6feLe3GsTXVjoge005b2+iZLMN8p4WNecAKGdnbHcZNP0nwT4d0i+06xHiG+uYdJZr22065uIvLR23HzThAzY3EgFioyDjpXUHTrqKNZIPJkmjupJljkYqrBs8ZAOCAeuDTbvS7mczqq2/l3FoIHG4rsIzjAwcj5vUYx3oAwZvh/4c1Kyu7GPULjz5tTbVo7iCePz7WfdgtGduAAQVwQepBqxqfw507U7u/kbUNRt4dVVF1S1gaIR3+0bcvlCykqMHYyZFb9xpa3AhhJjitYACkccYDKRxw38Ix/dAPvV5VCKFGcAYGTk/maAOfuPBGjXerXV7cxPIl1po0yS0YjyTACTjGM55x1qxoPh59DQRtrWp6lFHGIoI714yIUHYbEUt0HLljx15OdmigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAOetvBGiQNaPLbtdS2N/NqFrLM3zRSyli2NuAR85wDnseoBqjH8NtIh0OPToLm+je31GTU7a8V0M1vO7sxK5Tbj5mGGUjB5zXX0UAchL8NdIuNJms7q6vp5rrUItRur53jE1xNEwKbsIFCjaBhVAx0x1q5P4G0m6/4SMXLXEqeI0Ed4hkACgR7PkwARx6k810dFAHN2vgq1t73RLufUL67uNFgkgt5JjEN6OAvzhEUHAAAxj3zVaz+HOkWVvocMVxeldEvZb22LOmXeTfkP8ALyvznGMHpzXW0UAcX4d8JG0+I/iLxNcWklr9rZIbaNpVdXwoDzAKTtL4UYPOEGeuKnt/hxpFtpdpYpcXpitNWbVkJdNxmZ2cqfl+5ljx19662igDlZvh3o1xZ65bTPdOutXov5XLqGgmAUK0RC/LgopGc8+3FO1PwjaXOk251jXdQe60+4+1QatM8KTW7gYyAIxEBtJBBTBzzk811FULzTBcXKXUMvl3MXMZkXzUBwR909OvVSpPckcUAcneeC/Dy2MemzX9xcXs2qW+qTztPGLm5lWRdruAuNvAGFVRgYGKafBfh7UW1GKS+uEubzWV1mSAXEe8SQsFXA258s+WM9Tyea6WXTrx7tgFtmga4S4MrMQ4K7QVC4x0HB3e2O9Ni0q685UkS2SGO4kuFkRzvYtuwCu3A4brk9OlAGefD1pqWttq2naxf6Re3lvEb63spYiLhB90uHRipxld6bTjvwMdSBtUAZ4GOTmsv+x3nhtIruZfLto1CiJMPu24JEnUDn+HafftWqBgADt6mgAooooAKKKKACiiigApGXcpU9CMUtFAHOWvgbSLfwK/hOTz7nTXR0bzZMSEMxbO5QMEE8Eegpmm+BrKzurm71DUNQ1m7uLT7D59+8YaODnKKIkRRnOScbjxzxXTUUAcWnwzsodJ0aytdb1aB9ELrYXKG3MkUbJs8vDRFCAOhKlv9qpdM+G+m6Zb+H4hqGo3I8P3Es9o07xZYyKylX2xjIG44xg+pNdfRQBzFz8PtDurnxHPLFJ5niOBYL4gr91UKgrx1wc855qS78D6XcaFpOmW73Fj/YpRtPubYoJbdkXaCNylTlcggqQc9K6OigDkH+G2kTaX9knub6WV9Ti1S4vWdPOuZ42BUuQm3HygYVQMDjFWtf8AAWi+JtbXUtZjkuGWxksfIJHllHZWLdMhwVGCCMV0tFAHM2fge2t7vRrq61XU9QudHhlhgmu5Iy0iyAA7yqDcQAADwfXNQ2Pw60mwh8PRw3F6w8P3Etxa73T52kDBg/y8j5zjGO1dZRQB5Pq/gLULjVI9M0i01S10061FqbtLdWxsItsgkdkVcTlmx91htDMSOgNehah4dtNS8Q6VrM8ky3Gl+b5KowCN5i7TuBGTx0wRWtRQBzmreGrO48RR6rbaxd6LqVxELZ3s3hBu0UlgjLKjglctgqAwBPNc14j8FaPrV7qc1rrupxC4tk+2x2VzA4ka2I2Bmkidg3IzhhnHIOTXbppXkXkk9rPtEwxKJU8xiOcbXzuHJ6EsAOABVJdJvpLd4pktI9tm9rGY3Zt2cYY/KMDg8c49TQBWs9M0u08SXfiKzvZL69uYrfTrgechVNjHBIVeGy5yOnoBRZeFLOfxRF4judYu9XurVZYbbzhbhbYMfmUGKNGOMYw5bH15q+umXUs3nTJbwsDCqpE5ZdiPuznaOeemOPWrMWmf8TA31zKGn6DyV8tdozgNg5fg/wARI7gCgC/RRRQAUUUUAFFFFABRRRQAVyeq+HNT1rxzouoXn2KPTdFkluIWjkczTO6FApTbtUAE87jn0FdZRQByFr8NtKtby3YX2oSada3jX1tpUkiG2hmJJ3DCCQgFiQrOVBPTgYjb4Zaa810G1XVTY3moHUbrT/Nj8meXcGAJ2bwoKj5VYA45zXZ0UAc5rXgyDX7lRqOrak+nCZJm0sPH5Ejqdwy2zzcZAO0OF4xjHFZlx8L7GbWDfQa5q9oh1Iap9khNuYhcAY3ZeJnII7Fsc8Y4rtqKAOXuPAWnz3fiCVby+hi8QxeXfW8bR7C2wJ5ikoWDbRjGdvtTNa8A2uu6DBod3q2oppEdulvJZItuVnVcYLM0RcHgcoy+2DzXV0UAYi+FNPXxHLrIafzpbAae0W8bBECTxxu3c9c1iaD8L9P8P6tp97bazqs8em2kllbWlwbcxJA5BKfLEGPIHJYnjknmu2ooA5jTPCGn+G9LuLT+1b3+xfLdI7G5ljWC1RychWCq+OSBuZsdsVmR+D9E0uHw1pw1a7un0C5WawhkmhDqjKYgGCoCyANwep7sa6u80wXFyl1DL5dzFzGZF81AcEfdPTr1UqT3JHFVpdOvHu2AW2aBrhLgysxDgrtBULjHQcHd7Y70Ac1a+DNAlvmiOoXItUvpNRGgmeIwmZXP73GzzMbxu27tue1S/wDCCaFrFhqls95do+sX0GsXMSTJvikXYVC/LwhMYHIJPPPpvRaVdecqSJbJDHcSXCyI53sW3YBXbgcN1yenSpf7HeeG0iu5l8u2jUKIkw+7bgkSdQOf4dp9+1AGrRQBgADt6migCPbP/wA9I/8Av2f8aNs//PSP/v2f8akooAj2z/8APSP/AL9n/GjbP/z0j/79n/GpKo6zqX9kaPcXwtbi7aFcrBbQvK7noAFQFj74B4zxQBa2z/8APSP/AL9n/GjbP/z0j/79n/GuM8E61/bGteILaa41OZlMLN9qs7m1WMtGNyoJFXYM9AOcc85zWt4J3roEsTz3E4gvrqFHuZ3mfYszqoLuSxwAByTQBu7Z/wDnpH/37P8AjRtn/wCekf8A37P+NSV594gSa++JD2Uln4gv7ZdOhcJpOrNZpAzSSAu4FxFuyAOm4/L09QDvds//AD0j/wC/Z/xo2z/89I/+/Z/xridV8Z6hZ+LJNM021mmgspIYpol0m6uXn3gEkXCfu4tqsD8+7ODnb1pmoeLPEkN9ftZw6WbSz1iHTkilWTzJxKI+S4OEKmTP3X3AYwvWgDuds/8Az0j/AO/Z/wAaNs//AD0j/wC/Z/xrlrbxPeW2ia3JrVzYpc6XeG1FzHbSCOQlUZSIQzuzfvAAisSxGBjPGJNr13r3hSGTUNrTW3iG1gMi2Utn5gE0bAmGUl0OGxgk5xnoaAPRNs//AD0j/wC/Z/xo2z/89I/+/Z/xrirPxrqN74yNlBaSvYC9eyaNdIuspsBBmN1/qcbhjZjPI+bPFdzQBHtn/wCekf8A37P+NG2f/npH/wB+z/jUlFAEe2f/AJ6R/wDfs/40bZ/+ekf/AH7P+NSVwt+19pHjhr7W11RtOuJoo7K+sr1zBbg4Xyp7bO3ls/vdrn5xkpgUAdttn/56R/8Afs/40bZ/+ekf/fs/41yEvjC5Xx7a6XbzwXFlNcNayImn3H7pxGz5+1Z8otlcGMDcM9eKxbPxLdaH4XsLXTwUuL7Ur8Cb+zbi/EapPIT+5gw7E5A6gDqfQgHpO2f/AJ6R/wDfs/40bZ/+ekf/AH7P+NcdfeLdZt9E0fU2shY20yM9/Jc2E8nlEdFKLh4lbk+YysEA+Zea1/Gl1LB4B1m6s5njlSxleOWCQqynaSCrDGPY0AbW2f8A56R/9+z/AI0bZ/8AnpH/AN+z/jXI6JBe2Xhq7u9M0/VNPvQquE8Rao14soUZO0/aJBGCMjORg8kECnN41lbwXd+JYrYCDKx21tsaSVX3+WfMEZbPzH7qgnA6knAAOs2z/wDPSP8A79n/ABo2z/8APSP/AL9n/GuKtPGOrHw/cSXFt5l4L2GztrmfTLjT4pTKVAcxTHeApYg/N82OCM8a0mqazpl/oljqb2NzJfXckMs9vC8asghdwVQu205XHLNxzxnAAN/bP/z0j/79n/GjbP8A89I/+/Z/xriLrxD4h1LTbOXR72wsZW12WwdpbNp1eNHkVePMU5+QZwee2Ktpr97a63qYdI3trfVLe2uGzISqSQphlBYhcOy5AAGMnrzQB1m2f/npH/37P+NG2f8A56R/9+z/AI1xet+M9QtIrcWaLEb+6mS1nGmXF9shiAyxhhO5izZwQVAUgn0PUaBqE+q6BZ3t5ayWk80eZIZImjZT0PyuAyg4yARnB5oAubZ/+ekf/fs/40bZ/wDnpH/37P8AjUlFAEe2f/npH/37P+NG2f8A56R/9+z/AI1JRQBHtn/56R/9+z/jRtn/AOekf/fs/wCNSUUAR7Z/+ekf/fs/40bZ/wDnpH/37P8AjUlFAEe2f/npH/37P+NG2f8A56R/9+z/AI1JRQBHtn/56R/9+z/jRtn/AOekf/fs/wCNSUUAR7Z/+ekf/fs/40bZ/wDnpH/37P8AjUlFAEe2f/npH/37P+NG2f8A56R/9+z/AI1JRQBHtn/56R/9+z/jRtn/AOekf/fs/wCNSUUAR7Z/+ekf/fs/40bZ/wDnpH/37P8AjUlFAEe2f/npH/37P+NG2f8A56R/9+z/AI1JRQBHtn/56R/9+z/jRtn/AOekf/fs/wCNSUUAR7Z/+ekf/fs/40bZ/wDnpH/37P8AjUlFAEe2f/npH/37P+NG2f8A56R/9+z/AI1JRQBHtn/56R/9+z/jRtn/AOekf/fs/wCNSUUAR7Z/+ekf/fs/40bZ/wDnpH/37P8AjUlcHpNpfaH43A1/7VOdTurj7DdR61cyxAfM4je1bEaYQEAqG+72zQB3G2f/AJ6R/wDfs/40bZ/+ekf/AH7P+NefXr65o+ui8urbU5LmXU0iS7GpD7DJBJKFEYt/M4cIevlDlSd/qeIG1vSNWvdTmt9UlkW5jNldRakEsliZkUQvb+YMuSWGfKbJYYcfwgHoO2f/AJ6R/wDfs/40bZ/+ekf/AH7P+Ncb4lW4tfFNjdxnWLVTcwI1+bsmyCFtph+zo5yzE43vHgFs7+AK7agCPbP/AM9I/wDv2f8AGjbP/wA9I/8Av2f8akooAj2z/wDPSP8A79n/ABo2z/8APSP/AL9n/GpKKAI9s/8Az0j/AO/Z/wAaNs//AD0j/wC/Z/xqSigCPbP/AM9I/wDv2f8AGjbP/wA9I/8Av2f8akooAj2z/wDPSP8A79n/ABo2z/8APSP/AL9n/GpKKAI9s/8Az0j/AO/Z/wAaNs//AD0j/wC/Z/xqSigCPbP/AM9I/wDv2f8AGjbP/wA9I/8Av2f8akooAj2z/wDPSP8A79n/ABo2z/8APSP/AL9n/GpKKAI9s/8Az0j/AO/Z/wAaNs//AD0j/wC/Z/xqSigCPbP/AM9I/wDv2f8AGjbP/wA9I/8Av2f8akooAj2z/wDPSP8A79n/ABo2z/8APSP/AL9n/GpKKAI9s/8Az0j/AO/Z/wAaNs//AD0j/wC/Z/xp0rrFC8juEVVLMzdFA715x4RvbweJdNe9n1KP+0LeVmnurqaW31JuGRoEb5YcLuO0iNsdFcDcAD0XbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSigCPbP8A89I/+/Z/xo2z/wDPSP8A79n/ABqSvPPE3i5o/EdnbsutWcVvqcMCrFpl3susn5iZFj2MuOAoJzySOmADvwJsjLoR3wh/xqSiigBMj1FGR6ioqKAJcj1FGR6ioqjuJJIraR4ITPIqkpGGC7z2GTwKAC3sLS1vLq6gjCzXbK07bid5Vdo4JwOB2pbKytdPheKzQRo8rysNxOXdizHk9ySaxNJ1zU79tTs7nTLWDUrDaNkV60tu5ZdyjzTErA+o2cZB5zTNE1vWr7XLuw1XStPtktY1Lz2WovcAO3IjIaGPB2/NwTgEetAHTZHqKrLYWiapJqKxgXcsKwPJuPKKSQMZx1Y8471JXO3Hi5Lb4g2/hia0YLcWnnpeeZ8u/LYjK46lUYg5/hIoA073w7pd/qseo3EcouUUKTFdSxLKoOQJEVgsgGTgODjJ9ac/h/SpPO32wPn3aXsn7xvmmTbtbr22Lx046Vnad4qgv9V1m2MLQ2+lFQblmyJc7g2FA4wyMvuRVa/8ZQP4fk1DRdzyQ3kFtLFeWssDxmSRFO6NwrA7XyMjHQ80AbFz4b0m7tb2Ca3Oy+nFxOUmdGMoCgOrKwZGGxcFSMYqGDwlolvC8UdvIUkuI7lw91K+6WMgq5yx+bKgk/xEfNmll8S6XFrS6U88n2osEO2CQxq5GQjShdisRyFLAnIwOa1KAKA8O6WuttqqxyrcuQzqt1KIXYDAdoQ3ll8AfMV3cDnitTI9RUVFAEuR6ijI9RUVFAEuR6isifwxpNzqw1GeOd5d4cxG8l8guOjmDd5ZYYB3Fc5AOcitKs3TdX/tDVNVs/I8v+zp0h378+ZujV84xx97HfpQBCfBuhHUjfi3mWf7QLldt5MEjl5yyIH2oWyd20Ddk7s5qWfwto1xpa6e0EkcCTtcRtBdSxSRyMxZmWRGDqSWboRwSOnFIviPTX10aQJJhdsG2FrWVYnK/eVZSvlsw7qGJGDxwax/Dvjqy1QJb3zmO8ku57ZWitZfI3JK6KhlwUDkJnaWzz05FAGzfeFtH1Gwgs7uKYwwgqBHdyxtIp+8sjKwMit1ZXJDHkgmr99p9pqOlz6deRh7SeIxSRqxXKEYIyCCOPSsqPxbo0usf2alzL5/mmESG2lEDSDrGJivll/9kNnIIxxS2vizRb29vrS1vfNm0/cLsLE+ISpwVZsYDegzkjkZFADrPwlpVlBJBHJqM0UpTdHd6tdXC/KcgYkkbA9QOCODkVZbQNKaTUXa1VhqYAvELsUlwu3JTO0HGASACcDJ4FZNx460a30G51YLqEsNrMsE0K6dOJ0dsYBiZA4GGByRjB47VPeeMtGsBB9qku1aeET7F0+4Z4oz/HKoQmIdeXC9D6GgC1H4X0dNHm0uSKa5tZ/9Z9ru5biQ46fvJGZxg8jB4PIxTJPCejy6fDZyJdMsEvnRzG/n88PjGfO3+YePl5bkcdOKbc+K9FtNVtdNlvQ15eRiW3hijeUyof4xtByvHJ6AcnGasW+u6ddw2Etvcb01HP2XCNmTCljxjK4APXHp1oAiPhLQ/wCxTpUdoYLPzzcKtvPJE0chYtuR0YMhyT90jrjpVhtB0x4dQieDcmpY+1ZlbMmECZznIO1QMjHTPXmsnxZrutaBa/bNN0mwvrVSiuZ9ReBwzOFACiFwRyOcj6VNJ4iOkafDJ4ohjtLuZ2CW+mma+yo/iG2JXwB1OzA4yaAL+oaBpmp6bBY3Mciw25UwNBcSQyREDAKyIwdTjjg8gkHgmrtpbQWVrHb2wKxxjA3OWJ9yxJJJPJJJJPJrJv8AxRpOnLaNPcSS/bF3wLa28lwzJx8+2NWIQZGXOFGRk81lT+O7DTvEWo2WpTAW1vDbzRPb28kzBZAxLvsDbYxtHzkBR3NAHYZHqKMj1FYGqeL9F0e6a1vbqT7QkAuDFBbSzt5RJG/EasdoIOT0HGcZFTf29Zi+kie4t1gjslvTMZCMRkkbjldoXAzndn270AbOR6ijI9RXODxdp09klxaO4BuYYCt5bzWxPmOFVgrR7iD2ONpx94ckWpPEemRa5HpMksy3Up2oTbS+UzYLbPN2+XvwCdu7PtQBs5HqKMj1Fc9D4z0KfUlsYbyR5muGtSVtpfLWZSQY2k27FbjgEgnqM5rcoAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUVl2vhzSrPWH1OGGQ3TbtvmXMkiRbvveXGzFI899gGe9X6wU1zU4det7PU9JhtrW9d47WaO88yUsqlv3kewKgIU8q79vWgC9/wjek/22urGGRrpCWQNcyGJGPV1iLeWrnJ+cKG5PPJpZvDmlXGspqc0Uj3CMHVDcyeTvHR/J3eWXH98ru4HPArIXxZcNdi4GnRnRDdmy+3fav3vm7/Lz5WzGzf8u7fnvtxTLjxjPDfXkn2C3/sewultLm7kvdkyyHaMrDswVBdeS4J5wp4yAbFz4Z0m71iPUriKV543WQR/apRCXHRzCG8suP7xUngc8CtbI9RXKXviy5t7q6mt9Mjm0jT5/Ivbs3W2RGwCxSLYQ6ruGSXU8NgHHPTUAS5HqKMj1FRUUAS5HqKMj1FRUUAS5HqKMj1FRUUAS5HqKMj1FRUUAS5HqKMj1FRUUAS5HqKMj1FRUUAS5HqKMj1FRUUAS5HqKMj1FRUUAS5HqKMj1FRUUAS5HqKMj1FRUUAS5HqKMj1FRUUASMEdSr7WVhgg8gisbS/CWi6NeC5sIpw6ArEk17NNHAD1EUbuViHbCAccdOK1TnadoBOOATiubstc8QP4mi0rUdG0yKNommkmtNUkmaFc4UsrW6D5jkD5ux9KAOpyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUUZHqKiooAlyPUVXvLC01DyPtcYk+zzLPF8xG116Hg89eh4p9YGq69qOlatbI+lQyabcXEdsJxd/vy791h2EFR3JcHAY7cDkA6XI9aKiH3h9aloAhoqXYPSjYPSgCKobyJ5rKaKJYXZ0KhZ13I2R0Yeh6Vb2D0pkjRQxPLMyxxopZnZsBQOpJ7CgDlfC+g3+iw6gy2Wm6WkyqLXStPmd7WBgDlgTGmCxPIVB0zySa1vD+lvpGiw208gmumzLczDP72Zjl257ZJx6DAq1pmr6VrdmbvRtRtNQtgxUzWlwsqAjqNykjNN0vWtH1tZW0XU7LUVgfZKbS4WURt6NtJwfY0AWq5LX/B1xrmsX11HfCxaW0t0tLmIbpbeeKR3D7TwR8wGM88g12WwelV/tdl/aH2H7TD9s8vzfs/mDzNmcbtuc4zxnpQBy0nhbULS21SPRLuO2e4soLe2kLsHDR79xZsHaW3feGSCScZFZVl4J1mLTtTtZWtlW9vLS8QyalcXbxmNo98ZklUswxHkNxndjauMn0FJ7aW5lt45o3mhwZYlcFo88jI6jPbNRahfWel2v2m+k8qLzEj3bS3zOwVRgZ6kgUAcdP4Im/4TGW/VBc2dzdpeO0ur3cfkOoXgWy/upOUBBYrjPRsDPbVLsHpRsHpQBFRUuwelGwelAEVFS7B6UbB6UARVysVj4l0vxNq1zp1hpV5ZajcRzb59RkhkjxGqEbBA4P3c/eH4V1+welRRTW00ssUM0ckkDBZURwTGSAQGHY4IPPY0AcLa+FNfTxhBq95cR3QgvJX3SancEPC6soCwFfKiZAQPlBLcksMnL7HwvrqW9vpN9/Z39mxai1+bqGZzMf37SpGIym3uuX3+uF713mwelVdPvrPVLX7TYyebF5jx7tpX5kYqwwcdCCKAOTj8NayFi0dzY/2PDffbRdieT7QwEvmiPy9u0fNxv3ngfd54tJ4e1O18Oa1a6fdQ299fXk1xFMjFcB2yAW25VtvG4A46jOK6zYPSjYPSgDzyw8Ga1b6Prto5tlbUHhubfzdSuLsxyIFBjeSVSzLmMYf/aI2DbzqXOm+JI7+bUNOt9LafUbWOG7hnvJAts6bvmjYREyj5z8pEecdRnjr9g9KNg9KAOW0Twu+jazbTrKk0FtpMVgjtnzCyMSTjsDx3qj4W0ph4u1m680S2NlO9vYrsx5bSFZJxnuN5A46YIrt9g9KNg9KAMTxNpU+taDLZWrRpI8kTgyEhcLIrHoD2BrK8YeFrnWb6y1Gw3Sz2sckLW/9q3OnrIjlST5sGWyCo4KkHPbrW1qvijw5oNwkGua7pmmzOu9I7y8jhZl6ZAYgkVa0zU9M1qzF5o1/a6hbFiontZ1lQkdRuUkZoA5oeHtV0e+0278PRafIIbEWE9rczyxqiBgyukm12JHIKsPmyDuGOZL/AMN312fEhEtvu1awjt4jlgFdUdSWGDgZYdM11uwelVob60n1C5sYpN1zaqjTJtI2h87eehzg9KAMF/D902sahdiSHy7rSY7JBuOQ6mQknjp849+vFUJ/BdzeW81vPcwxpNoceml0yxWRSTuwQMryO/NdtsHpUQmtzdG2EsZuAgkMW8bgpON2OuMjGaAOXu9L1/WdItoNUj022nt722mxbTySK6xyBmbJRSCQOFwcf3jnjMvPCniG48YQ6s1xDcR2uoC4iEup3Cq0JVkMf2cL5SMobIf5ixGDtzkd3PNbWqK9zNHCrOsatI4UFmOFUZ7knAHepdg9KAORtvDF7Dp0du0sBdNbfUCQzY8szM+On3sHp0z3rqal2D0o2D0oAioqXYPSjYPSgCKipdg9KNg9KAIqKl2D0o2D0oAioqXYPSjYPSgCKipdg9KNg9KAIqKl2D0o2D0oAioqXYPSjYPSgCKipdg9KNg9KAIqKl2D0o2D0oAioqXYPSjYPSgCKipdg9KNg9KAIqKl2D0o2D0oAirlNE0/xFH4kmvte0/SpTIXRLyLUZHeGHOVjSJoAFBwu758kjJJwAOw2D0qhba1o95qk+mWmqWU9/bDM9pFco0sX+8gOV6jqKAOa/4RnVSV0gmzOhi++2/aDMxuD+980Q+WU243/wAe/p/D3pde8Jya34gWWTStFWIlN+q7M3ojU5MIBQ8HGN3mDhiNvc9L/bGk/wBs/wBkf2lZ/wBp7PM+xfaF87b/AHtmd2PfFI2taOutLo7apZDU2XeLI3KecVxnPl53Yx3xQBzWoeG9WmOoaXaGyOj6rcGe4mllcTQBseYiR7Cr7tp+YsuNx4OOevAAAA6CqtzrWj2eqQaZd6pZQX9yMwWktyiyy/7qE5boegq/sHpQBFRUuwelGwelAEVFS7B6UbB6UARUVLsHpRsHpQBFRUuwelGwelAEVFS7B6UbB6UARUVLsHpRsHpQBFRUuwelGwelAEVFS7B6UbB6UARUVLsHpRsHpQBFRUuwelGwelAEVFS7B6UbB6UARVlaDpdxp8V1PqMiS397cNNO8ZJUDOERcgHCoFH1ye9bWwelUbTWNJv9QubGx1Kzuby0OLi3huFeSH/fUHK/jQBZoqXYPSjYPSgCKipdg9KNg9KAIqKl2D0o2D0oAioqXYPSjYPSgCKipdg9KNg9KAIqKl2D0o2D0oAioqXYPSjYPSgCKipdg9KNg9KAIqKl2D0o2D0oAioqXYPSjYPSgCKuT1fw5fX/AIpt721sdJtykkbNrEcjpeiNWyYNoT5lbGDmQDDfcOOex2D0qhJrWjw6zHpEup2SalKm+Oya4UTOvPITO4jg847GgC2PvD61LSbR6UtABRTd496N496AHVn69qGnaVoN5f62YxYW0Zlm8xQwwOeh98Ve3j3o3j3oA8/t9U07W/DvibW7e7s9XmurMRy2Gj3azNDGFYKrOm7Mh3MSQMDoA23LHgW/jv8AxTO8WqadryppsUS6hpciGOFVY4ikVQR5h3ZyGwQDhI+/oG8e9G8e9ADq818URXVr8S5tf0uzmvbzSdMt3NrbjMlxC8soljUdzgBgO5QV6RvHvRvHvQB51ayv4dk8VajfJK11LZ29zc+S4VldzJxvI+VVGF3Hoq57Vi2OrG88MaxZPeRTCDVtPkt401aXUQIXkhIdJ5VV3QsHweVyGAPBx6/vHvRvHvQB5pd6lq4+Jr2899Z2kovYhaQz6zNG81rtXcEs1jKTZ+f5yxKkdVC16bTd496N496AHUU3ePejePegB1FN3j3o3j3oAdXDab4n0HRvG3ie01jW9OsLma9hMUN1dxxPIDbxgEKxBPPHFdvvHvRvHvQB5lZeJvtPxOt2juTbwNdz2U8cusyO+QrbFe02+VDllyhB3sOecnFPw9c2kRs003WLt/EB1mcHTTeOB9mN1J5jG2yF2Bdx80qTkD5jwK9Z3j3o3j3oA81t9SX+14ims3b+K21Ty5tKa+kKrbebg/6NnasYiwwk2jsd3PNuyuZtO0nxTr9/qOp3nlXlxBDb/aSsdvGHxhey88l2yVGcYAxXf7x70bx70AePQXkWreB/E9jeX7PDZ3NvcxC1125u9sBEZMi3DbJHjyHOeVBVgD8vG3qt/oi3wXUfE11Z6YmnxNos0OsSoLtzu3MJA+bl+E+Vi+QRwd3Po28e9G8e9AHCadFquseL9MGs6hf2ph0SG5ubGCYxJJOXIJcLg8c/KMA9DkDFTeHLvUbvxEui3E90R4f8wXUsjN/pRfiDLfx/uyWPX5sV2u8e9U7DT4NPnvZomleW9n8+V5Gyc7QoA9AAoAFAGT4+/wCROuP+u9v/AOj0rG+JF/e2d3pqtcW9ppLpKZri51qXS41mG3yw00aMem8hTgMRznGK7vePejePegDz2+1BVvNCg8Za5LYWkmnB0ura8ktIrm8BAIMg2EnbysZ4bJO07eK+uWTTap4tv01HUoZLLTLaeDyLh7fMixyEO4TaW90bK+q16VvHvRvHvQBwerxXGqeJNTim1PUYILfQorlIrS8eAeaWl+fKEHPyjjOD3BwMR3Gra0Y7mXT555bv/hGIbiKNRvzMS2XCdC34c4Ar0DePejePegDz57/TU8O20vhXxBeahE+qWSTSHUJLjaWmUOpdiSpIOGjzgf3RnmlqHigL8TbTZdG1ji1EWV0JtYkDBTGwXdZhfKRGfaFkZgzHAGcgV6dvHvRvHvQB5zYC6Jg1WTVdSkn/AOEkltFja8fylhM7J5flg7WGBwWBI7EDAr0im7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOrg9J8Q+H/ABN4utYNM1XSYo9KlmW2sobiP7TPLhkdvLBykYy3GMseeABu7rePejePegDzcXduLNNCE0X/AAkv9um5+zbh54X7TuM+3g7PJ43dMfLntV6/8Q+H9W8Wr4dj1XSbFrO/jluI5LiNLi4uQQypHHkE843Pg5+6AeSvdbx70bx70AeU+J9RiTxFq9oLu0jea9tmk0S4lVbvUNmwrJbkqSAcYxtcHacNEcmvWByPSm7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQBX1Nbh9JvFsTi5aBxCQej7Tt/XFcH4I1djfaTpUM+jX6QWhSW3soWFzpLBQSJnLnJZgQcpGSRnBwceibx70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOopu8e9G8e9ADqKbvHvRvHvQA6im7x70bx70AOopu8e9G8e9ADq4HxTf2x8UWWnfb9OmJv7eY6NHEYr2VwwxPv3HfGoAJwgGEI3jBFd5vHvRvHvQA6imhwadQBDRRRQAVDeGQWM5hkMTiMlXERkKnHUKOWPt3qao7iEXFu8ReSPeuN8bbWX3B9aAOEs/FV5p/h/xBetqE+oxWAT7M+rQLa3CswxmWNUjKR7iMFkUkBjyMGtjwnqQubm6tZddvtSnRRI0WoaabR05wWjHlx7os9Dhv941atfCltHbX8epXt5q0uoII57i7MauUUHaoESIqgZJBAByc5qbSvDw06+kvbvUr3Vbto/KSe88oNHHnOxRGiDGeckEn1oA164nVPEuo6X8ULazmnQaE9pEkyMqjy5pXcRvu64ymzHTLCu2rG1LwppWs3F++qQtdRahaJZz27t8jRqzMMYwQcsec9hQBk6J4kvrzUtfuLpozYQRRz2MZKxgRHeNzOezbN2T0BH45kvjV/EHhK8uLKezjms9UtLd5tJ1H7VC4aWInbKFQn5XwQQO45rp77wnp9/DfxSNNGl9BFAwjK4jERJQqCCMgnocg4HGM1UTwRb+Xdrc6rqV2byW3nnaZ4vmlhKkONsYC5CKCowuBwASSQClc/EjT7fxidE8zTv3d0lnIr6iq3RlcAgpb7SWQbgC24HrwQM12dZDeHgNabULXU760jlcSXFnC0YhuHAwGbKFwcAZ2MoOOc8516ACiiigAooooAK4q51q/svHjW+s6heaZZyyxx6eptY3sbtSOVaXaXSbduABdAflwG5rtaxb3w2NQ1HzrnVtQaz8xJW03dH5DOpBBzs8wcgHaHC8dMZFAFG48aR2/jq28PONOY3DFAqakrXaHYX3Nb7eEIH3t2eRxWJB44GgeFrV766s3u7zULyKKTVtRFrCFSZ/vSsGIwAAAFPYcAEjeXwRbR6p9si1TUY1W+N9Fbq0XlxysCHxmPcwYMQdxbGfl28VK3hC2TTYbayv76ymt7iW4hvIGjMsbSOWcfMjIVO4jDKeMdwDQBnT/ECJdN0S/tre2NrqgJM9zerDEhGBsSTaVeRjnaCVDYJ3Ctnxbf3OmeDNWvrKQw3NvZySROFDFGCkg4IIP60zVfDI1bSU0+XV9SijKNHcPG8Za6RvvK4ZCoz6oFI6KQOKvalpNvqmhXGkzGSO2uIDAxjPzKpGOCc849c0Acn4V8QXAh1e/u9XvtT0ixt1d5tRtI7e4imClpE8tY4yF27CNyA88Eg8JpXxMttT0vWLmNNNuJdOsjfLHp2qLdKyYbCyMqjy3yvK4Yc8E10beG7NtUkvd8oE9r9lurfK+VcqOFLgjO4AkAgjgkHPGM3UPDF3D4P1jT7XUtQ1SW4spYLWG8kjxHlCFQFVXPYbnLN6t1oAY3jDULPzW1bRYrZH0+W+tPKvTK0gjALJJ+7Ajb5l+6XHXnjmpffEKTTdA06+1Ox0/SrnVGJtodU1VbeNYwobMsuwhW54VQ3UcjnGla+DYmtZBqmoX97JLZNaJ57R5tI3UB1jKoM5wPmfceBzV6/8PRXtjZwwXt3YXFiALe8tinmoMbSMOrIQRwQVI78EAgAxrrx/Gmj6JqVpbWzW2qZ3T3V6IYY8cbVl2srOxyEB2q2PvCmeJfiPYeHNa/s+Z9PVooUnuEu9RSCYq5IAhjwTK/yk4yo6YJJra1Lw5/aekpYPq+owjY0c0sbxlrhG+8rhkK8+qhSOilRxSXXhmOS8huLDUb7TGSJIZVtGjxcRr91H3oxGMkblKtyeelAGbrPjHULC+1SHT9Fju4dKt47q4nlvfKDRsGJCAIxLgLwDgHP3h0q1BrmsT+Np9NWwsjpcdlFcif7Swny5cY2bNp5XGNwx1yc4Fu68NWd22rGSScf2tbrbz7WHyqqsoK5HXDHrmmXXhi2utSa7N3dxrLZ/YriCNl2XEfzY3ZUsCNzYKlevOaAMK18fzalb+II9Ni0e5vNLtTcxi11X7TEfv4WUqgMbjYcqAw9G71bt/EOrtNpCX1lEl5eafcXJtrW63wsU8sqNzRBsnd2wBk/e4NT6f4JtrCV3k1TUbwSWX2CSOdogjQjO0YSNQpUMwBXGc/NuwCLen+GYrGfTp5dQvL2bToJYIZLgxgsjleG2IoOAigH88nmgCjfeOrKy0dNTMW62/s/7dKzShPKDEKiknjLMSMkgDaSam8IeLYfFUF3sfT3mtJFSRtMvxeQHcMjEoVcnqCCowR34NWYvC2nR2OoWbebJDfyM7hnwY8ksFQgAgBiSO4J61a0nTZtNt2S51W91SRiP3155YYAdBiNEX8cZPcnAoAv0UUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFcfb6nfJ4/SxGrXk8Unm+fbXtktvCgAyotmKK0p/vEPIAMk7cgV2FY0fhtRrkeo3OqahdrAzPbWk7xmK3ZhglSEDscEj52bGTjFAHNT/ABF0268Z6bYWHiHTEgN89pNbC5iMsrBHySpO5QHVVHQk57EZsXer6qYNS1+HUpI7XTb1rcaasMZjmjRwrlm2l95JYjDAD5cqec9Te6VBf3+n3czyLJp8rSxBSMMWRkO7jphj0xzWdc+EbW61J52vbxLWWZbifTlZPInlXGHbKlxyqnCsFOOQcnIBQ8RW+qvrdrb6R4o1K1ubxwy2kcVo0MMSY8xzvhL45A+995h0HTrRwPWqUOlQQ61dapuke4uIkiO8grGi5ICjHGSxJ65NXaACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooA5nSdSu7O28RtqF5NqC6ZdP5TSpGr7BCkm35FUHBYgHGfUmq2lXur2l/os2papJfR65Gd9uYY1S1k8syDyiihtuAQd5Y9DnrnorTSrezmv5ELv9vm86VXIIB2KmBx0wo9e9Z+leFLfS7+K5+33t2lrGYrK3uWQpZoeCE2oGPAAy5Y4HXk5AN2iiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigArBuJ72Dx7ZQLfTNZ3VlM7WjJHsV0aPDA7d+TvOQWI6cCt6qkumQzaxbakzSCa2ikiRQRtIcqTnjOfkHf1oAuD7w+tS1EPvD61LQAUUUUAFR3Ewt7aSYpJJ5altkS7mb2A7mpKhvBIbGYQxmVyhAQSmMtx0DDlT7/yoAz9F19dYmuraXT7zTby0K+da3nll1DDKtujd0IOD0bPHIFP0rX7TWL29t7JZSLMoDKwASUMDhkOckcEZIGe2Rg1zeheHby40/WrWWLVdCsr8IsQubqK4vVbGHcylpdwI2gB2cgAjgYFW/C/hzU9F8TaxPe6re31pNHBHbG5S2UMFU54ijTGM4HAGPXrQB1lYsnirT4vGkXhiRZlvprQ3UblR5bKGI27s53cE4x0B9K2q4jxJ4W1bVPE13qWktDa3cNnbNp13PygnjklLKwHzbSr4PHRuKAOisfEdhqGq6pYwM4bSyouJXAWP5gTwc84wQfQgjsaq3ni/TR4fGr6Nc2ur2xuYrffaXKumXkVD8y5GRuzj+VZH/CP6tpNvrf9kwJPPNZW6RSS7G8+UNIZW2sQNxLkjdhSSM8ZrJtfD/iObTtYhvLS/mlvNQsbyKe+ltPMZEaLeriHagZRGTgAggrhmOQAD0CTWNMi1aPSpdRtE1GVC8dm06iZ1/vBM5I4POKuV5/ceGNWHjeeR5NXmsLvUIr5Wtns0toyiqMSl08/I2cbCQQQPl5r0CgAooooAKKKKACuYvfHEFjq13ayaRqUlrYzRw3WoRiEwwtIFIyDIJCPnXJCHGa6evPtQ8Iai3iHWNdt4JJp1voLi2spLnMF7GkSAgxltiuGBKuQGDKvOKAO0utZ0ux1C2sL3UrS3vLvIt7aWdUkmx12KTlvwp41XT21VtMW+tjqCxiVrQTL5oTpuKZzj3xiuH13w3q994pnuU/tj7FqaW5eOzkslWBo+dsxlVpAM8gxFv4uM4J05dL1KH4gRXOnWNxHZzS+deyySQPbPiIoHUH98k3Cr8o2bc5JJoA6OLWNMn1WbTINRtJNQgUPLaJOpljU9CyA5A5HJHemWuu6RfXd3a2OqWVzcWRxdQw3CO8HX74ByvQ9cdK4zQvDGsaf4mAu5NVnitbm5ureZ5LNbMmUsf4EFwW+fBDfLwTuOAKj8P6R4nXWRd61a3rb9OmtCshs1it3JDARLFhvKOMLvLPxyF6kA7QeJNDNjc3o1nTza2u03E/2pNkO4BhubOFyGBGeoI9ao3njHS7O90/zLu0/s29tJboak10ohVEKAHd90g+Z1z275rBsvClzoXgjwpBpukI0mitHPc6bC8atK/lsrlWJCFwzbslgCR1rU/sm4v8AxVoOrS6OtpFbW90zxyNGzW8shTbnaSN5AfJUkcnk5yQDpYLqC6tY7m1mjnt5EDpLGwZXU8ggjgj3rK0TxImuSboNMv4LR13215MqeVcrnqu12Zexw4UnPAODh3hWwuNM8NW9pdRCKaNpCU3AgAyMRyMjoRXLWHhu+/tm+/sfS7/wzaXUE6XcT3cT2s8zjCywRo7FDnkttjyDypPQA6v/AISfR5LS+nsdRtL77A2y5S2uoiYmzjaxLAKfZiKsy6zpkGqQ6ZPqNpFqFwu+G0edRLIvcqmckcHkDtXINpuq3/w9n0dvDX9n3tvZJaR/v4WWYqV/1TK2fLO3Pz7D0+Worzwvqx8Z3EjS6vNYXt5Bdj7K9mlvEY1QBZS6efwUyPLJBzj5eaAOxl8QaNb30dlPq9jFdyymGO3e5QSPIADsCk5LYIOOvIqm/jLQBfajYQ6rZz6hpsTS3NktxGsqADJ+8wAxkZJIAzyRWHN4XvGi8RvHYR+fqGsW9zG4KBpYo/J+YnPbY+AeeOOtSa3peqzyeLLS202WZNWsP9GuVljCeYIinlsCwYMTyDgrjqRQB1H9r6cL63sXvrZL24j82K1aZPNdB1YLnJA9RkVXsPE2iapq95penaraXN/YttubaOUGSM8dV645Az0zx1rnb/R9Vi8SaXc6XY3IZxAt7J5kD2rxx/8APVX/AHiyLlthi6nG44GBsaRb3tn4p1sT2MgtbyVLmG8DoYziJEKEbt4bKk/dxjHOeKAN6iiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKxIPEofWo7C50rULJJ2ZLW7uFjEdwygkhQHLqcAn51XOK264/R5NXv/FLXXiDw7qduYnkS0kae2a1t05AYBZi7O46sU4zgYGSQDRi8XW82pGFNPvfsQuDajUyI/s5mDbdn3/M+98u4ptzxnpUjeJkbWnsLTTNQvIoZBDc3sCIYbeQgHawLB2PIyUVgM8kYOOf1Pwsb/xNHHbaJdWkP2xLue9Ook2bBHD/ACW4k/1zFRljGMZY7yeta48Oaxa63cR6bBqwmmvzdW+pRaiI7OFHcM6SweYC5++OY3zuX5h/CAdLeeLraz1KS3NhfTWsEqw3WoRqnkWzsAQr5cOfvLkqrAZ5Iwcb9cPqOkav5Or6Da6bJNa6xdNMNRWWMR26Pt8wOCwkLDDbdqsDlckc47cDCgDsKAFooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAAkAEnoK5/SvF9vquow2y6df20V1G0tldzrH5V2q4yU2uWXgg/OqkjpW7MJDBIICol2nYXHAbHGfavPvCfh/VbHxBZXEmlXentDDJHqEtzLbyW9wzck2yoxaIFwGICxKRyyluQAeiUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFZra0sfiaPRpLK5Vprdp4ro+X5ThSAyjDbwRuXqoHPBNaVY93Y3MvjLTb5I820FpcRyPuHyszRFRjrztb8qANiiiigBm8+1G8+1MooAfvPtRvPtTKraisTabcLPs8toyGEkpjU5HQsOQD0zQBc3n2o3n2rgPDNnYaU/iOy1Gys9FgSOKW4stOlxZwRlD8yuAh3MFJY7E7cH7x1PB+jw2M97fabpcWi6ZdrGLbT4YhEDt3ZmaNQAjNuHHXCruwflAB1e8+1G8+1MrzvXtETWvijKk3hvQ9djj0yDd/az48gGWXJjHkyZJxyPl6Dn0APR959qN59q891fXPEKeMLm00m2vha6a1uPLiNmls8b4LNKZXEoAG4L5YAyv8AFyBFqWr+IxfalPa62IYrXXbewgtjaRshjkEQbeT8zYLkjaykHqWHFAHo+8+1RT3kNqivczRQqzqimRgoLMcBRnuScAVxkGvXmnaLr0eoahd3k1lqIsradIIjcSl1jKqFAWPduk2gkBRwWzg5yF1O/wBR8KhdUkupJrTxJbW4a9EHngCaM4fyP3eQWI+XsBnnNAHp+8+1G8+1efWOveIrzxnKVt79dPh1F7KWNzZrapGB8rff+0eYTtIGNpDcL0au+oAfvPtRvPtTKKAH7z7Ubz7UyvItZ06GbxzrN3daXYxxLqlpGdewGurA+VEQFG0FUJ+Xdv4L8qRmgD2DefajefavPte17xEvjC5ttJt74waekEhSI2a28qucu0zTOJQuAQDGBgg53dBpNqupWnjwQ313K2m3Uot7VYVheBXEZYpJ0lWXIJzym3HAJoA6/efajefavPdB8ReIb3xEbi8t76PTWuri3mE5s1t4VjLbfL2uZy/yDIYYwSQAMU3w/wCJ9Sv9fugZ9ReyuNMe9tvt0Vqi4DAK8KxEyBCD0m+bge9AHom8+1RNeQpcx27zRLPIrMkRYBnAxkgdSBkZ+orz3S9T8Rnwh4cfV9e/03xBLEJbyG2iRbRWhLBYgVILMVAy4YZY4AGFFvUbO+n8YeH7SLXZPPWzvUkv44YjMQGiHTBjD5xk7COD8ozwAd5vPtRvPtWJ4WvbnUPC1rcX83m3BDpJNtClyrsu4gcAnGeBiuO0WKTwfqEyS6VYXtxd2k9xa6zp77ZNQCLvxcp1Z+eHy4OTjZnBAPTN59qN59q89uru9Hw3utUm1+PV2vbFLpI5LeBo03FT+7AXmMZx8+89Pmoudf8AEcvjO6hsrW/FnYXUEDKDZratE6qWkkLuJt3zNt2ADKgYPNAHoW8+1G8+1efXWqa99r1m9TWXjt9N1iG2htEtotrxuIQyyMVLH77EFSpHckcU3Vvt8E3jm5bWru4htrLMWn3EcEkEe6DdnaY84yDwTg5OQaAPQ959qN59q4i51jU7DxFp0k1xINIlS3g22yQNHHK/GJlOJBuyoQx5Aydy4GaueG470+LPEj3GtX13BFdrHFaTeUY4QYo2+XCBh1Ixux3IJyaAOr3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamVwcdjb2fxHsb1NPsLRbx50jvbCfzZr87ckXBKqQq7TgAyDOBlcAEA7/efajefavML/SrnSdfivoLfR7i8utXVIdXWUm+dWk+e3KBPurGHB/eEAJnaD06aaztovihZ3UVtClxPpc6yzLGA8gWSLaGbqcZOM9M0AdTvPtRvPtXmHinSJ7LVNR1mK20e8umuYhZ6nLK326zkO1Vt0UIcg5z99QRI2VPO70wdOetAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2rjdCtl0yHxdHotrDA0d7I8MMMYRfMNvG3QYHLHJ9zWfoVnpthqPhe70OKIXeqWzPqE6DEl2gi3GSUj77CQr8zcgsRnk0Aehbz7Ubz7UyigB+8+1G8+1MooAfvPtRvPtTKKAH7z7Ubz7UyigB+8+1G8+1MooAfvPtRvPtTKKAH7z7Ubz7UyigB+8+1G8+1MooAfvPtRvPtTKKAH7z7Ubz7UyigB+8+1G8+1MrmbuztY/idpl5HbQpczadcpLOsYDuqtFtBbqQMnAPTJoA6kOSRT6iH3h9aloAreaP7sn/AH7b/CjzR/dk/wC/bf4VZooAreaP7sn/AH7b/Ckdo5I2SSNnRgQytESCPQ8VaooAxLfQNCs9Jm0q00Ozg06fPm2cViqwyZ67kC4OfcUuk6DoegeZ/YWi2emedjzPsdiId+OmdqjNbVFAFbzR/dk/79t/hUYjt1umuVt8TugRpRAdzKCSFJxnAJPHuau0UAZN3pWk39/bX19pdvc3doc29xNaB5If9xiuV/CpGsrBt+6wjPmTLO+bb70gxhzxyw2jB68D0rSooAy59O026tbq2udNhmgvDm5iktdyznAGXBGG4AHPoKjh0bRrZCtvpFrEpMZISzCjMf3Dwv8ADgY9McVsUUAZJ0rSW1hdWbS7c6kqeWL02g84L/d37d2PbNXfNH92T/v23+FWaKAK3mj+7J/37b/CjzR/dk/79t/hVmigCt5o/uyf9+2/wqvJZWEsdzHLYxul5/x8q1tkT8bfnGPm4AHPatGigDHuNF0a7uLOe60e1nmsf+PSSSyDNb9PuErleg6Y6CnnTNLbWF1ZtMgOpLH5S3htP3wT+7vxuxz0zitWigDJTStJj1h9Wj0q3XUpE8t71bQCZl4+Uvt3EcDjPaorXQNCspBJZaJZ27hncNFYqh3OMOcherDqe/etuigDMm0/TbnSzplxp0MtgUEZtJLXdFtHRdhGMcdMUW9hp1otstpp8UC2sZitxFa7fJQ4yqYHyjgcDjgVp0UAU4EgtoRFbQGGNckJHCVAycngD1OapWGi6LpV5cXel6Pa2VzdHdcTW1kI3mOScsyqC3JJ59a2aKAMWDQ9EtY7tLXRrSFL1/MuljsgonbOdz4X5jnuc1JcaVpN3qdvqN1pVvNfWoxBdSWgaWEf7Llcr17GtaigDONnYssqtYxkTSCWUG2/1jjGGbjkjavJ54HpUU+laTdXj3d1pdvNcvAbdp5LQM7RHrGWK5Kn+70rWooAyZdK0q41C2v59Lt5byzUrbXD2gMkAPBCMVyo+lSLY6emqPqS2Ea3zxiJ7oW371kByFL4yRntnFaVFAFbzR/dk/79t/hR5o/uyf8Aftv8Ks0UAVvNH92T/v23+FHmj+7J/wB+2/wqzRQBW80f3ZP+/bf4UeaP7sn/AH7b/CrNFAFbzR/dk/79t/hR5o/uyf8Aftv8Ks0UAVvNH92T/v23+FHmj+7J/wB+2/wqzRQBW80f3ZP+/bf4UeaP7sn/AH7b/CrNFAFbzR/dk/79t/hR5o/uyf8Aftv8Ks0UAVvNH92T/v23+FHmj+7J/wB+2/wqzRQBW80f3ZP+/bf4UeaP7sn/AH7b/CrNFAFbzR/dk/79t/hR5o/uyf8Aftv8Ks0UAVvNH92T/v23+FHmj+7J/wB+2/wqzRQBW80f3ZP+/bf4UeaP7sn/AH7b/CrNFAFbzR/dk/79t/hR5o/uyf8Aftv8Ks0UAVvNH92T/v23+FHmj+7J/wB+2/wqzRQBW80f3ZP+/bf4VRt9I0i01SfU7XSreG/uRie7jswsso9GcLk9B1PateigDKXS9KTWG1ZNLgXUnj8trwWgEzJ/dL7d2OBxntVgpA1wtw1vmZFKLIYTuVSQSAcZwcDj2FXaKAMqTTNLl1aPVJdMgfUIk2R3bWmZkX+6H25A5PGaueaP7sn/AH7b/CrNFAFbzR/dk/79t/hR5o/uyf8Aftv8Ks0UAVvNH92T/v23+FHmj+7J/wB+2/wqzRQBW80f3ZP+/bf4UeaP7sn/AH7b/CrNFAFbzR/dk/79t/hR5o/uyf8Aftv8Ks0UAVvNH92T/v23+FHmj+7J/wB+2/wqzRQBW80f3ZP+/bf4UeaP7sn/AH7b/CrNFAFbzR/dk/79t/hR5o/uyf8Aftv8Ks0UAVvNH92T/v23+FHmj+7J/wB+2/wqzRQBW80f3ZP+/bf4UeaP7sn/AH7b/CrNFAFbzR/dk/79t/hR5o/uyf8Aftv8Ks0UAVvNH92T/v23+FHmj+7J/wB+2/wqzRQBSjSCF5Xht/LaZt8jLCQXbAGTxycADPsKrWelaTp17c3mn6Vb2t1dndczwWgR5jzy7Bct1PX1rWooAreaP7sn/ftv8KPNH92T/v23+FWaKAK3mj+7J/37b/CjzR/dk/79t/hVmigCt5o/uyf9+2/wo80f3ZP+/bf4VZooAreaP7sn/ftv8KPNH92T/v23+FWaKAK3mj+7J/37b/CjzR/dk/79t/hVmigCt5o/uyf9+2/wo80f3ZP+/bf4VZooAreaP7sn/ftv8KPNH92T/v23+FWaKAK3mj+7J/37b/CjzR/dk/79t/hVmigCt5o/uyf9+2/wo80f3ZP+/bf4VZooAreaP7sn/ftv8KPNH92T/v23+FWaKAK3mj+7J/37b/Co2SBrhJ2t8zRqVSQwncoOMgHGQDgfkKu0UAQLICwG1+veNv8ACp6KKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAP/2Q==)

Now, following this there will be different analysis that were done on this data, starting with PCA.

PCA on IBM Data

# Principal Component Analysis

Principal Component Analysis(PCA) is used to analyze the continuous variables in a given dataset, which helps us in interpreting which all variables contribute more to the flow of data and in the end cluster the data in various groups.

Consider a 2D dataset(has 2 variables/columns) that has all its observations(rows) plotted on the X-Y axis.

We do a little preprocessing before we begin, i.e. we center and scale the data. Centering: Refers to subtracting mean of each column from each of its points. Scaling: Normalization

Now we plot the new point on the XY axis.

We try to find a line among all these points such that **minimizes** sum of square of perpendicular distance each point from that line.

**OR** Try to find a line that **maximizes** the distance of projection of each point onto this line from the origin.

This line that we get is the **Principal Component 1(PC1)**. It will have the maximum inertia(variance), or it kind of pulls the majority of the data.

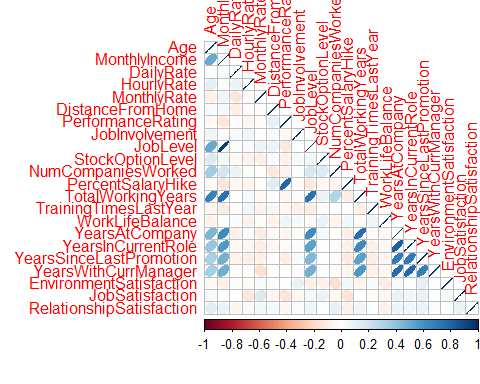
Next, **Principal Component 2(PC2)** is a line orthogonal to PC1(*We take the orthogonal line as next component in order to reduce any effect from the PC1, so we will just get the effect of PC2 on the observations*)

**Factor Score**: Instead of distance from XY axis, we take the distance w.r.t PC1 and PC2 thereby giving Factor scores of the observations. PCA gives one map of factor scores, from which we can make out the observations that have more effect on the components. For intrepretation in this graph we consider the distances between factor score.

**Loadings**: These corresspond to the variables/columns being plotted on the PC axis. Another PCA map gives loadings of variables. For intrepretation we consider the angle between various variables.

Based on boh these maps we find the important variables and factors that drive the overall data, adn divide it into various groups.

### Corelation Plot



From the correlation plot, we see that year related variables have significant positive corelation amongst themselves. Apart from the above variabels we have +ve correlation between Age-Monthly Income, Job Level,Num of COmpanies Worked. Also we see signinificant relation between Job Level-Monthly Income.

### Heat Map

dev.new()  
color4Var <- prettyGraphs::prettyGraphsColorSelection(ncol(new.ibm.emp.data))  
  
corrMatBurt.list <- phi2Mat4BurtTable(new.ibm.emp.data)  
col <- colorRampPalette(c("#BB4444", "#EE9988", "#FFFFFF", "#77AADD", "#4477AA"))  
corr4PCA.r <- corrplot::corrplot(  
as.matrix(corrMatBurt.list$phi2.mat),  
method="color", col=col(200),  
type="upper",  
addCoef.col = "black", # Add coefficient of correlation  
tl.col=color4Var,  
tl.srt = 45, #Text label color and rotation  
number.cex = .5,  
diag = TRUE # needed to have the color of variables correct  
)

A close up of a device

Description generated with high confidence

After the removal of qualitative data, the heat map shows,strong +ve correlation between Monthly Rate and all other variables. ALso we see similar trend with Monthly Income & Hourly Rate. Perfect COrrelation between Performance Rating and Percent Salary Hike.

## Create DESIGN variables.

### 1st Design Variable : Job Level

We choose Job Level as 1st design variable. Job Level is related to the position of employee in the company. In this data we have 5 levels for Job Level (1,2,3,4,5) With 1 being the position wise seniormost employess and 5 accordingly loewr ranked ones.

### 2nd Design Variable : Performance Rating

We choose Performance Rating as 2nd design variable. Performance Rating is related to the year end evaluation of employees. In this data we have 2 levels for Performance Rating (3,4) With 4 being the best and 3 the next best.

## Running the PCA

We run the epPCA and epPCA.inference model by passing the data containing only quantitative variables and also the design variables.

new.ibm.emp.data1 <- subset(new.ibm.emp.data, select = -c(JobLevel))  
new.ibm.emp.data2 <- subset(new.ibm.emp.data, select = -c(PerformanceRating))  
  
res\_pca\_d1 <- epPCA(new.ibm.emp.data1, center = TRUE,DESIGN = DESIGN$rows$JobLevel$color\_observ, scale = TRUE, graphs = TRUE)  
  
res\_pca\_d2 <- epPCA(new.ibm.emp.data2, center = TRUE,DESIGN\_1$rows$PerfRating$color\_observ, scale = TRUE, graphs = TRUE)  
  
# do inference PCA  
  
res\_pca.inf\_d1 <- epPCA.inference.battery(new.ibm.emp.data1,scale = 'ss1', DESIGN = DESIGN$rows$JobLevel$color\_observ, make\_design\_nominal = TRUE)

## [1] "It is estimated that your iterations will take 0.1 minutes."  
## [1] "R is not in interactive() mode. Resample-based tests will be conducted. Please take note of the progress bar."  
## ===========================================================================

res\_pca.inf\_d2 <- epPCA.inference.battery(new.ibm.emp.data2,scale = 'ss1', DESIGN = DESIGN\_1$rows$PerfRating$color\_observ, make\_design\_nominal = TRUE)

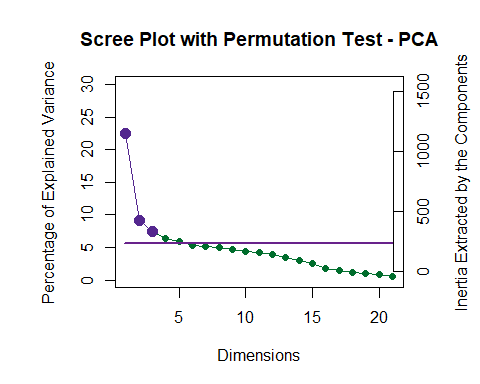
## [1] "It is estimated that your iterations will take 0.02 minutes."  
## [1] "R is not in interactive() mode. Resample-based tests will be conducted. Please take note of the progress bar."  
## ===========================================================================

### Scree Plot

We first do a scree plot with inference, which shows how many components are required to define the data, and which all out of these are the significant ones.

* We find that 3 components come up as significant during permutation test.

#Get the Eigen Values list  
EigenValues <- res\_pca\_d1$ExPosition.Data$eigs  
  
  
PlotScree(ev = EigenValues,   
 p.ev = res\_pca.inf\_d1$Inference.Data$components$p.vals,  
 title = "Scree Plot with Permutation Test - PCA",  
 plotKaiser = TRUE  
 )

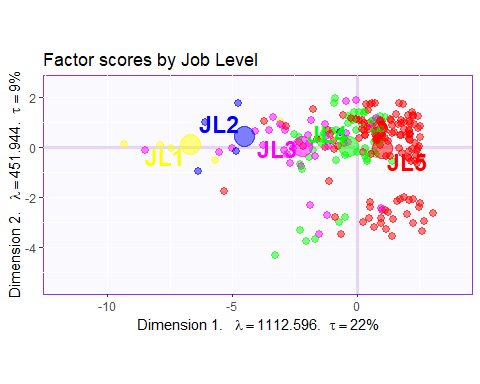


a001a.screePlot <- recordPlot()  
print(a001a.screePlot)

### **Factor scores Based on Job Level**

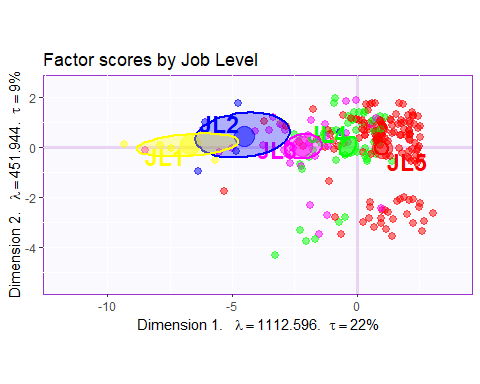
Factor scores are the coordinates of the 237 employees on the components. The distances between them show which employees are most similar. Factor scores (states) can be color-coded to help interpret the components.

#### **Between Component 1 & 2:**

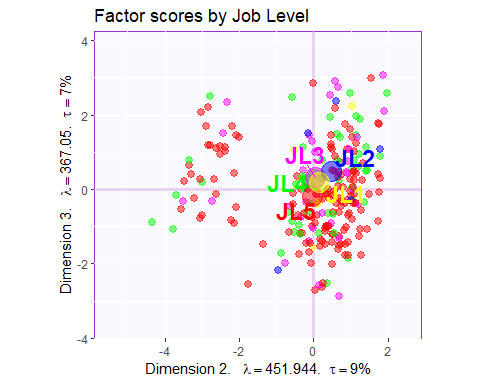


#### **Confidence Intervals**

# Confidence Intervals ----  
# Bootstrap for CI:  
BootCube <- PTCA4CATA::Boot4Mean(res\_pca\_d1$ExPosition.Data$fi,   
 design = DESIGN$rows$JobLevel$color\_observ,  
 niter = 100,  
 suppressProgressBar = TRUE)  
  
  
GraphElli <- PTCA4CATA::MakeCIEllipses(BootCube$BootCube[,1:2,],  
 names.of.factors = c("Dimension 1","Dimension 2"),  
 col = rownames(BootCube$BootCube[,c(1,2),1]),  
 p.level = .95  
)  
  
JL.withCI <- jl\_1\_2 + GraphElli  
  
  
print(JL.withCI)



#### **Between Component 2 & 3:**

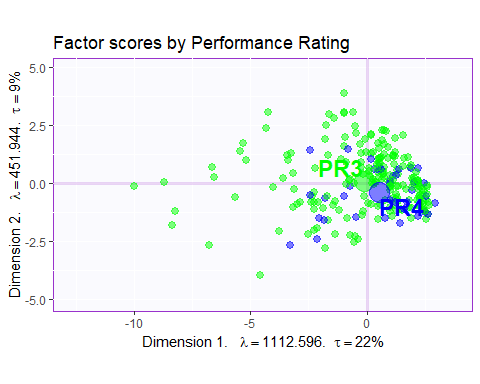


### **Factor scores Based on Performance Rating**

We plot, and color based on Performance Rating

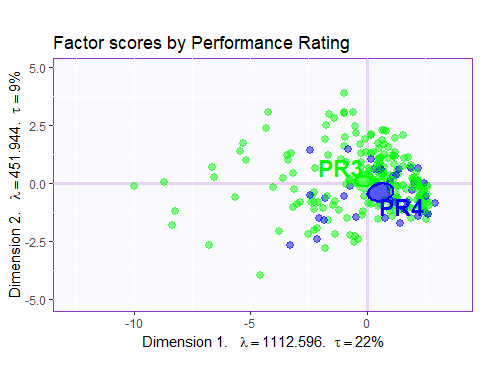
#### **Between Component 1 & 2:**

#Plotting Factor Scores for Dim 1 & 2  
baseMap.pr.12 <- createFactorMap(Fi.1, col.points = DESIGN\_1$rows$PerfRating$color\_observ,  
 col.labels = 'darkorchid',title = "Factor scores by Performance Rating",display.labels = FALSE,  
 display.points = TRUE,axis1 = 1,axis2 = 2)   
  
  
#Calculating the means for Dim 1 & 2  
data\_means\_PR <- PTCA4CATA::getMeans(Fi.1, ibm.emp.data$PerformanceRating)  
rownames(data\_means\_PR) = c("PR3","PR4")  
  
  
#Mapping the means  
MapGroup\_pr\_12 <- PTCA4CATA::createFactorMap(data\_means\_PR,  
 # use the constraint from the main map  
 constraints = baseMap.pr.12$constraints,  
 col.points = DESIGN\_1$rows$PerfRating$color\_groups,  
 cex = 7, # size of the dot (bigger)  
 display.labels = TRUE,axis1 = 1,axis2 = 2,  
 col.labels = DESIGN\_1$rows$PerfRating$color\_groups,  
 text.cex = 6)  
  
#Adding all the plots  
pr\_1\_2 <- baseMap.pr.12$zeMap + x\_y\_labels\_1\_2+MapGroup\_pr\_12$zeMap\_dots + MapGroup\_pr\_12$zeMap\_text  
  
print(pr\_1\_2)



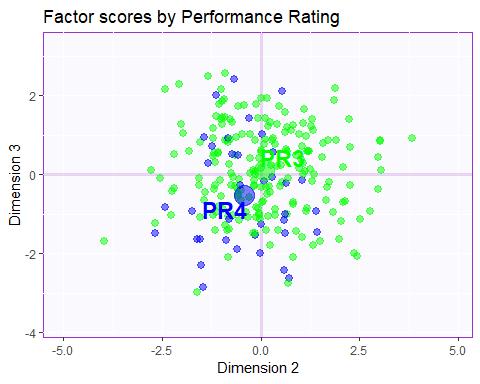
#### **Confidence Intervals**

# Confidence Intervals ----  
# Bootstrap for CI:  
BootCube <- PTCA4CATA::Boot4Mean(res\_pca\_d2$ExPosition.Data$fi,   
 design = DESIGN\_1$rows$PerfRating$color\_observ,  
 niter = 100,  
 suppressProgressBar = TRUE)  
  
  
GraphElli <- PTCA4CATA::MakeCIEllipses(BootCube$BootCube[,1:2,],  
 names.of.factors = c("Dimension 1","Dimension 2"),  
 col = rownames(BootCube$BootCube[,c(1,2),1]),  
 p.level = .95  
)  
  
PR.withCI <- pr\_1\_2 + GraphElli  
  
print(PR.withCI)



#### **Between Component 2 & 3:**

#Plotting Factor Scores for Dim 1 & 2  
baseMap.pr.23 <- createFactorMap(Fi.1, col.points = DESIGN\_1$rows$PerfRating$color\_observ,  
 col.labels = 'darkorchid',title = "Factor scores by Performance Rating",display.labels = FALSE,  
 display.points = TRUE,axis1 = 2,axis2 = 3)   
  
#Mapping the means  
MapGroup\_pr\_23 <- PTCA4CATA::createFactorMap(data\_means\_PR,  
 # use the constraint from the main map  
 constraints = baseMap.pr.23$constraints,  
 col.points = DESIGN\_1$rows$PerfRating$color\_groups,  
 cex = 7, # size of the dot (bigger)  
 display.labels = TRUE,axis1 = 2,axis2 = 3,  
 col.labels = DESIGN\_1$rows$PerfRating$color\_groups,  
 text.cex = 6)  
  
#Adding all the plots  
pr\_2\_3 <- baseMap.pr.23$zeMap +MapGroup\_pr\_23$zeMap\_dots + MapGroup\_pr\_23$zeMap\_text  
  
  
print(pr\_2\_3)

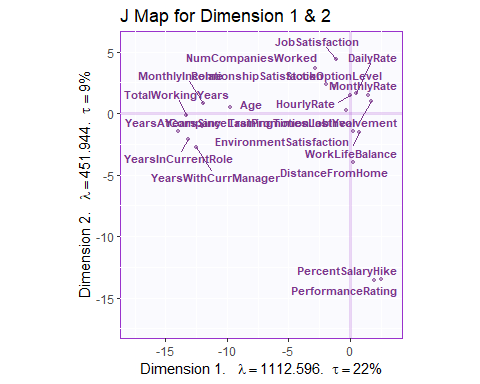


color4Var <- prettyGraphs::prettyGraphsColorSelection(ncol(new.ibm.emp.data))  
col4Levels <- data4PCCAR::coloringLevels(rownames(Fj), color4Var)  
col4Labels <- col4Levels$color4Levels  
  
varCtr <- data4PCCAR::ctr4Variables(Cj)

### **Loadings**

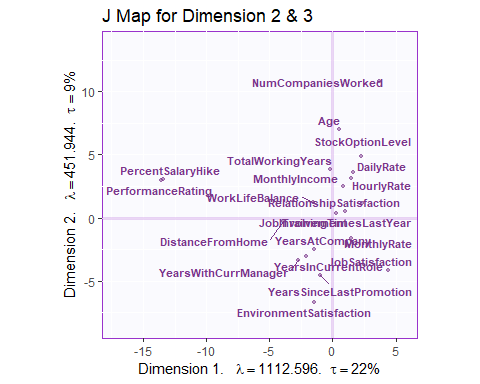
#### **Loadings for Component 1 vs Component 2:**

col4Levels.imp <- data4PCCAR::coloringLevels(rownames(Fj),col4ImportantVar)  
  
baseMap.j12 <- createFactorMap(Fj, #constraints = constraints.sym,  
 col.points = col4Levels.imp$color4Levels,col.labels = col4Levels.imp$color4Levels,  
 display.labels = TRUE,display.points = TRUE,text.cex = 3,force=2,cex=1,  
 title = "J Map for Dimension 1 & 2",axis1=1,axis2=2)  
  
print(baseMap.j12$zeMap + x\_y\_labels\_1\_2)



#### **Loadings for Component 2 vs Component 3**

baseMap.j23 <- createFactorMap(Fj, #constraints = constraints.sym,  
 col.points = col4Levels.imp$color4Levels,col.labels = col4Levels.imp$color4Levels,  
 display.labels = TRUE,display.points = TRUE,text.cex = 3,force=2,cex=1,  
 title = "J Map for Dimension 2 & 3",axis1=2,axis2=3)  
  
print(baseMap.j23$zeMap + x\_y\_labels\_1\_2)



## Inferences

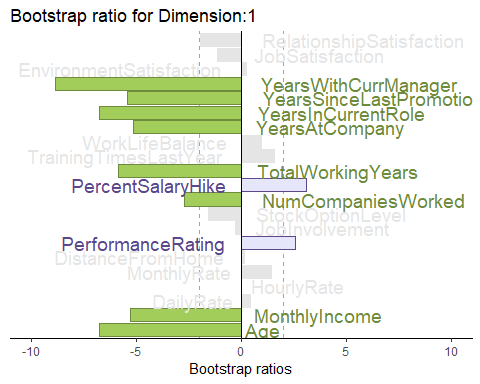
### **Bootstrap Bars**

We are trying to find out the significance of the different variables used in the analysis, by using bootstrap method to generate many samples from the same dataset.

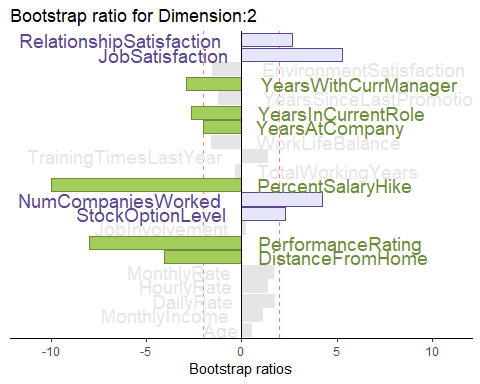
As we run the inference bootstraping takes place within and we get the bootstrap ratios for each variables. The plot will show bootstrap ratio of each variables for each component.

The ones greater than 2 will be shown as significant and the if the variable pops out in the top half it coressponds it is significant towards positive side of the component and otherwise.

BR <- res\_pca.inf\_d1$Inference.Data$fj.boots$tests$boot.ratios  
laDim = 1  
ba001.BR1 <- PrettyBarPlot2(BR[,laDim],  
 threshold = 2,  
 font.size = 5,  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios',horizontal = FALSE  
 )  
  
print(ba001.BR1)



#  
laDim = 2  
ba002.BR2 <- PrettyBarPlot2(BR[,laDim],  
 threshold = 2,  
 font.size = 5,  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios',horizontal = FALSE  
)  
  
print(ba002.BR2)



For PC1 :  
- Age,MonthlyIncome,Job Level, NumofComapniesWorked,TotalWorkingYears and other year realted variables are significant in -ve half of component.  
- Performance Rating and Percent Salary Hike are on +ve side of the component  
For PC2 :  
- DistFrmHome,PerformanceRating,PercentSalaryHike,YearsAtCompany,YearsinCurrentRole,YearsWithCurrMangr are significant in -ve half of component  
- StockOptionLevel, NumofCompaniesWorked,Job & Relationship Satisfaction are on +ve side of the component.

## 

## Conclusions

* Factor score when plotted between PC1 and PC2, colored by Job Level showed distribution along the PC1
* Factor score when plotted between PC2 and PC3, colored by Job Level didnt show any distirbution
* Factor score when plotted between PC1 and PC2, colored by Performance Rating, did not show any distribution
* Factor score when plotted between PC2 and PC3, colored by Performance Rating, did not show any distribution
* Loading for PC1 and PC2, Percent Salary Hike & Performance Rating were inversely correlated to Num of Companies worked & Relationship Satisfaction. Also, Year related variables are orthogonal to above 2 groups.
* Loading for PC2 and PC3, Num of companies worked & Age were inversely correlated to Environment Satisfaction & Year related variables. Also, in this case Performance rating and percent salary hike are orthogonal to above 2 groups.

MCA

# Multiple Correspondence Analysis

MCA is an extension to the CA, wherein we analyze the relationship between several categorical variables in a data table. Indicator Matrix - Matrix comprising of 0’s and 1’s. So, MCA is techincally a CA done on indicator matrix of a data table. Even quantitative variables can be analyzed by binning them,once binned these are again converted to binary values using disjunctive coding(one hot encoding).

**Design Variable: Job Level**

### **Binning the variables**

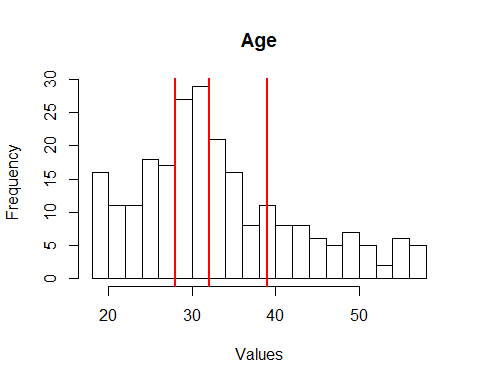
* We bin the variables in the data table to prepare for MCA, for that we first plot the histograms for the quantitative variables.
* First histograms are plotted, then we break it such that equal amount of data comes under each partition.
* The ordinal ones have been used as it is.

### 

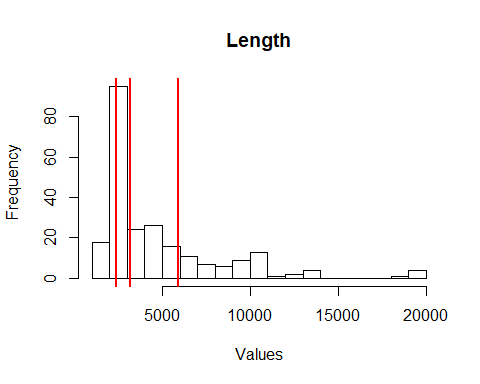
### **Binning using histograms**

We now plot some of the histograms for the binning process.

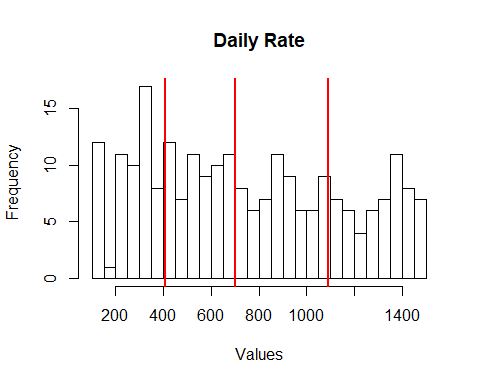
#Age ---- NOrmal Distribution  
  
hist(new.ibm.emp.data[,1],breaks = 25, main="Age", xlab = "Values")  
qts <- quantile(new.ibm.emp.data[,1])[2:4]  
abline(v = qts, col = "red", lwd =2) # plot the cutoff line



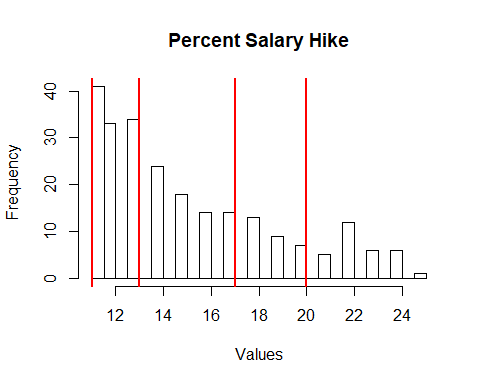
#MOnthly Income ----   
  
hist(new.ibm.emp.data[,2],breaks = 25, main="Length", xlab = "Values")  
qts <- quantile(new.ibm.emp.data[,2])[2:4]  
abline(v = qts, col = "red", lwd =2) # plot the cutoff line



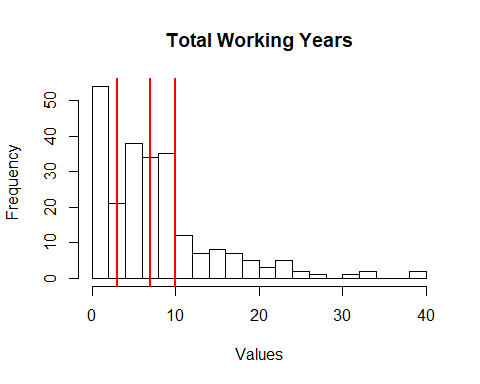
#Daily Rate ----   
  
hist(new.ibm.emp.data[,3],breaks = 25, main="Daily Rate", xlab = "Values")  
qts <- quantile(new.ibm.emp.data[,3])[2:4]  
abline(v = qts, col = "red", lwd =2) # plot the cutoff line



#Percent salary hike----  
  
hist(new.ibm.emp.data[,13],breaks = 25, main="Percent Salary Hike", xlab = "Values")  
abline(v = c(11,13,17,20), col = "red", lwd =2) # plot the cutoff line



#Total Working Years----  
  
hist(new.ibm.emp.data[,14],breaks = 25, main="Total Working Years", xlab = "Values")  
qts <- quantile(new.ibm.emp.data[,14])[2:4]  
abline(v = qts, col = "red", lwd =2) # plot the cutoff line



## Running MCA

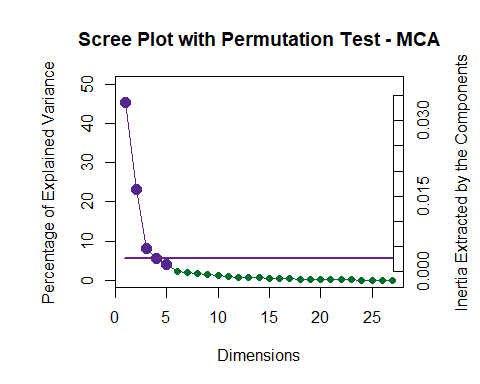
# Running the Symmetric MCA  
resMCA.sym <- epMCA(new\_df, symmetric = TRUE,make\_data\_nominal = TRUE,   
 graphs = FALSE,DESIGN=ibm.emp.data$JobLevel)  
  
# to run a plain MCA but asymetric  
resMCA.asym <- epMCA(new\_df, symmetric = FALSE,make\_data\_nominal = TRUE,   
 graphs = FALSE,DESIGN=ibm.emp.data$JobLevel)  
  
#Inference MCA  
resMCA.inf <- epMCA.inference.battery(new\_df, make\_data\_nominal = TRUE,   
 graphs = FALSE,DESIGN=ibm.emp.data$JobLevel)

## [1] "It is estimated that your iterations will take 0.13 minutes."  
## [1] "R is not in interactive() mode. Resample-based tests will be conducted. Please take note of the progress bar."  
## ===========================================================================

### **Scree Plot**

We get close to 5 components showing significant inportance in the permutation test

EigenValues <- resMCA.asym$ExPosition.Data$eigs  
  
  
PlotScree(ev = EigenValues,   
 p.ev = resMCA.inf$Inference.Data$components$p.vals,  
 title = "Scree Plot with Permutation Test - MCA",  
 plotKaiser = TRUE  
 )

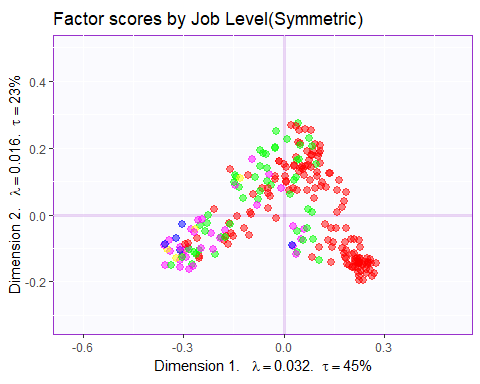


a001a.screePlot <- recordPlot()  
print(a001a.screePlot)

### 

### **I Map for DImension 1 & 2:**

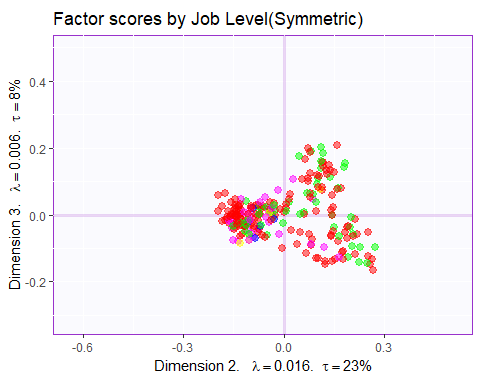
baseMap.i.12 <- createFactorMap(Fi, constraints = constraints.sym,  
 col.points = DESIGN$rows$JobLevel$color\_observ,  
 col.labels = 'darkorchid',title = "Factor scores by Job Level(Symmetric)",display.labels = FALSE)   
  
FS\_JL\_12 <- baseMap.i.12$zeMap + labels4MCA\_12  
  
print(FS\_JL\_12)



### 

### **I Map for DImension 2 & 3:**

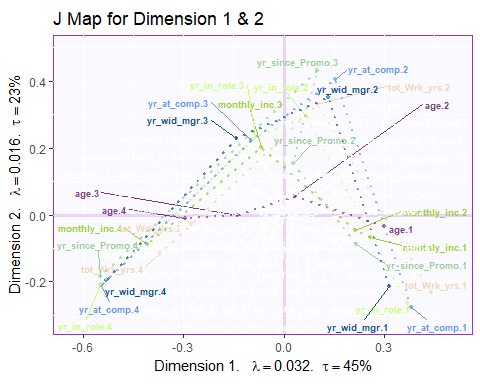
baseMap.i.23 <- createFactorMap(Fi, constraints = constraints.sym,  
 col.points = DESIGN$rows$JobLevel$color\_observ,  
 col.labels = 'darkorchid',title = "Factor scores by Job Level(Symmetric)",  
 display.labels = FALSE,axis1 = 2,axis2 = 3)   
  
FS\_JL\_23 <- baseMap.i.23$zeMap + labels4MCA\_23  
  
print(FS\_JL\_23)



### 

### **J Map for Dimension 1 & 2:**

baseMap.j.12 <- createFactorMap(Fj, constraints = constraints.sym,  
 col.points = col4Levels.imp$color4Levels,col.labels = col4Levels.imp$color4Levels,  
 display.labels = TRUE,display.points = TRUE,text.cex = 2.5,force=2,cex=1,  
 title = "J Map for Dimension 1 & 2")  
  
  
lines4J <- addLines4MCA(Fj, col4Var = col4Levels.imp$color4Variables, size = 1)  
  
Loadings\_12 <- baseMap.j.12$zeMap+labels4MCA\_12+lines4J  
  
print(Loadings\_12)



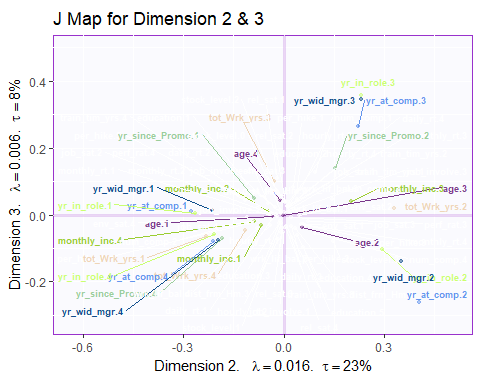
### 

### **J Map for Dimension 2 & 3:**

baseMap.j.23 <- createFactorMap(Fj, constraints = constraints.sym,  
 col.points = col4Levels.imp$color4Levels,col.labels = col4Levels.imp$color4Levels,  
 display.labels = TRUE,display.points = TRUE,text.cex = 2.5,force=2,cex=1,  
 title = "J Map for Dimension 2 & 3",axis1 = 2,axis2=3)  
  
  
lines4J <- addLines4MCA(Fj, col4Var = col4Levels.imp$color4Variables, size = 1)  
  
Loadings\_23 <- baseMap.j.23$zeMap+labels4MCA\_23#+lines4J  
  
print(Loadings\_23)

## Warning: Removed 1 rows containing missing values (geom\_point).

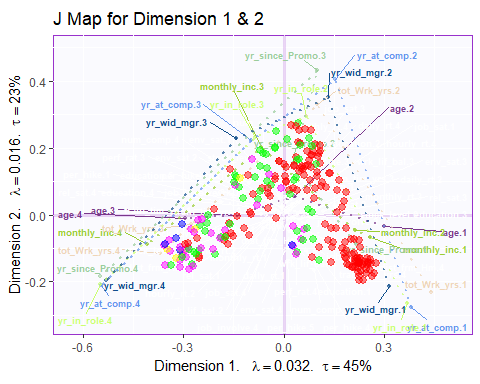
## Warning: Removed 1 rows containing missing values (geom\_text\_repel).



### 

### **I&J combined for Dimension 1 & 2:**

print(Loadings\_12+baseMap.i.12$zeMap\_dots + baseMap.i.12$zeMap\_text)



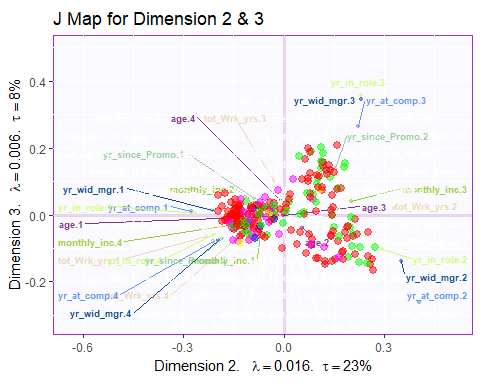
### 

### **I&J combined for Dimension 2 & 3:**

print(Loadings\_23+baseMap.i.23$zeMap\_dots + baseMap.i.23$zeMap\_text)

## Warning: Removed 1 rows containing missing values (geom\_point).

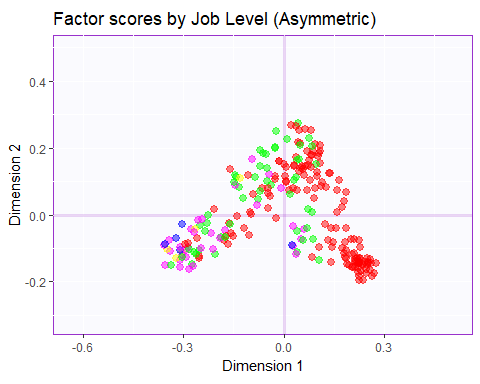
## Warning: Removed 1 rows containing missing values (geom\_text\_repel).



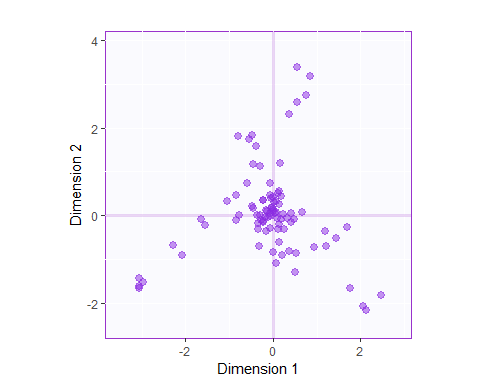
### 

### **Creating Base maps of Factor Scores Asymmetric:**

baseMap.i.a <- createFactorMap(Fi, constraints = constraints.sym,  
 col.points = DESIGN$rows$JobLevel$color\_observ,  
 col.labels = 'darkorchid',title = "Factor scores by Job Level (Asymmetric)",display.labels = FALSE)  
#dev.new()  
print(baseMap.i.a$zeMap)

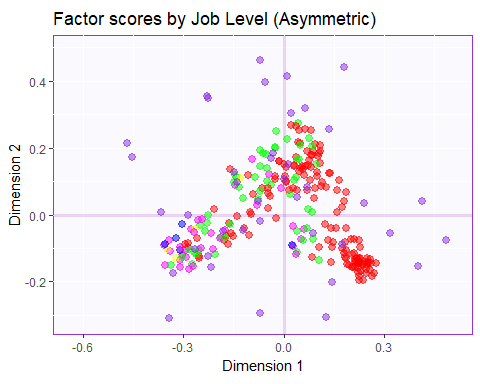


baseMap.j.a <- createFactorMap(Fj.a, constraints = constraints.asym,  
 color.points = 'darkorchid4',display.labels = FALSE)  
  
print(baseMap.j.a$zeMap)



print(baseMap.i.a$zeMap + baseMap.j.a$zeMap\_dots + baseMap.j.a$zeMap\_text)

## Warning: Removed 48 rows containing missing values (geom\_point).



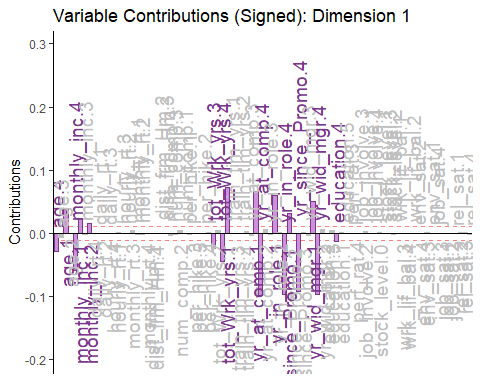
### **Plotting observation for symmetric and asymmetric:**

symMap <- createFactorMapIJ(Fi,Fj,  
 col.points.i = DESIGN$rows$JobLevel$color\_observ,  
 col.labels.i = DESIGN$rows$JobLevel$color\_observ)  
  
asymMap <- createFactorMapIJ(Fi,Fj.a,  
 col.points.i = DESIGN$rows$JobLevel$color\_observ,  
 col.labels.i = DESIGN$rows$JobLevel$color\_observ)  
  
labels4CA <- createxyLabels(resCA = resMCA.asym)

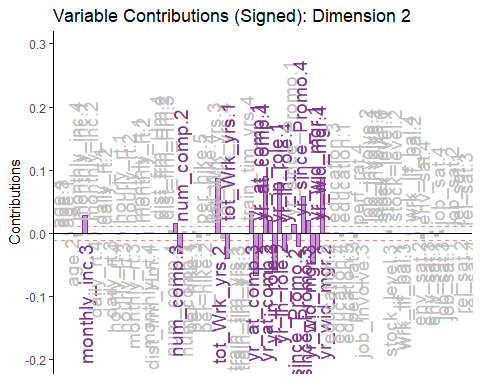
### 

### **Contribution Plot:**

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Contribution Plots ----  
# get the Contributions and make a plot.  
  
signed.ctrJ <- resMCA.sym$ExPosition.Data$cj \* sign(resMCA.sym$ExPosition.Data$fj)  
  
b003.ctrJ.s.1 <- PrettyBarPlot2(signed.ctrJ[,1],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(resMCA.sym$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 1',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
  
print(b003.ctrJ.s.1)

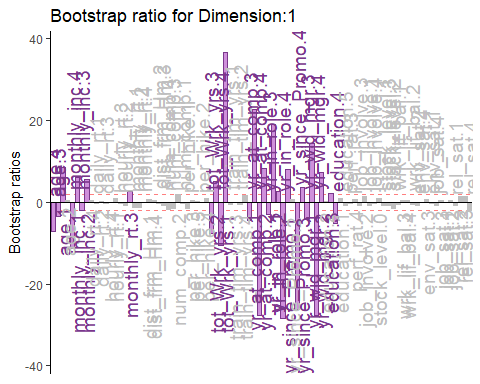


b004.ctrJ.s.2 <- PrettyBarPlot2(signed.ctrJ[,2],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(resMCA.sym$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 2',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
  
print(b004.ctrJ.s.2)

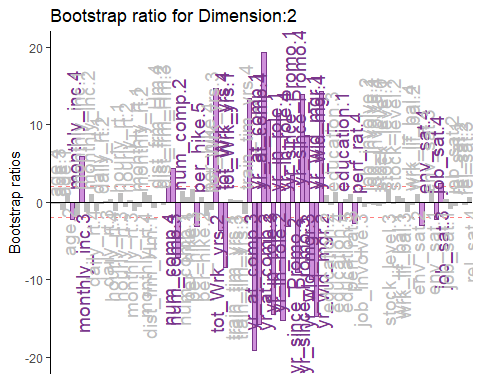


### **Bootstrap Bars:**

BR <- resMCA.inf$Inference.Data$fj.boots$tests$boot.ratios  
laDim = 1  
ba001.BR1 <- PrettyBarPlot2(BR[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resMCA.sym$Plotting.Data$fj.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios')  
  
print(ba001.BR1)



#  
laDim = 2  
ba002.BR2 <- PrettyBarPlot2(BR[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resMCA.sym$Plotting.Data$fj.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios'  
)   
print(ba002.BR2)

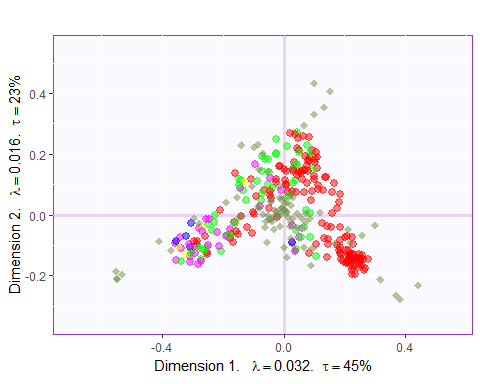


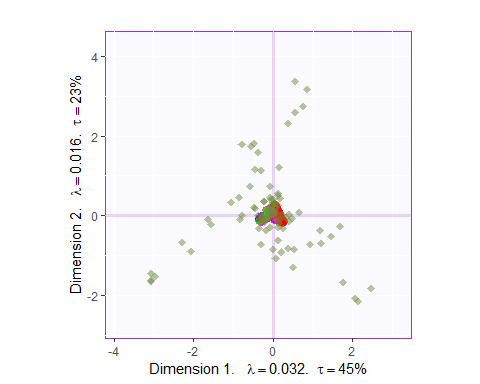
**Dimension 1:** -Monthly Income\_3,yr\_at\_comp(3,4),yr\_wid\_mgr(3,4) are some variable levels on +ve side of the component -Monthly Income(1,2),yr\_at\_comp(1,2),yr\_wid\_mgr(1,2),num\_comp\_4 are some variable levels on -ve side of the component

**Dimension 2:** -yr\_at\_comp(1,4),yr\_wid\_mgr(1,4) are some variable levels on +ve side of the component -Env\_Sat\_4,job\_sat\_4,education\_1 are some variable levels on -ve side of the component

### **Symmetric and Asymmetric Plots:**

# draw the maps ----  
map.IJ.sym <- symMap$baseMap + symMap$I\_points +symMap$J\_points + labels4CA   
 #symMap$J\_labels #+ symMap$I\_labels +   
print(map.IJ.sym)

   
map.IJ.asym <- asymMap$baseMap +   
 asymMap$I\_points + asymMap$J\_points + labels4CA   
  
  
print(map.IJ.asym)

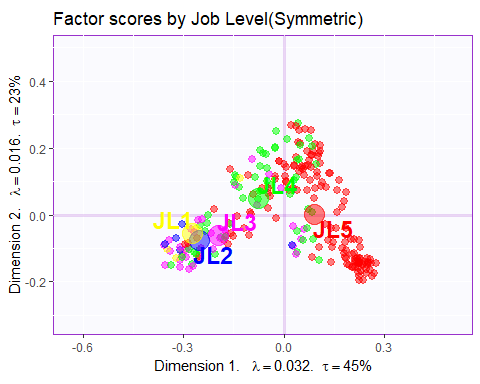
 **Symmetric plot when colored by Job Level gives kind of horseshoe shape.**

## Inferences

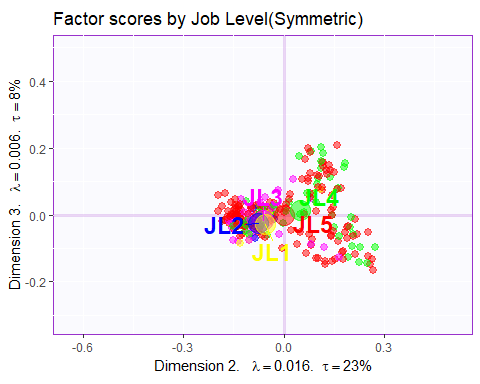
We now plot the contribution and bootstrap ratios

### **Means for Dimension 1&2, Dimension 2&3:**

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Inferences  
  
#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Mean Map  
# create the map for the means  
# get the means by groups  
  
data\_means <- PTCA4CATA::getMeans(Fi, new.ibm.emp.data$JobLevel)  
rownames(data\_means) = c("JL5","JL4","JL3","JL2","JL1")  
# a vector of color for the means  
col4Data <- DESIGN$rows$JobLevel$color\_groups  
col4Means <- unique(col4Data)  
# the map  
MapGroup12 <- PTCA4CATA::createFactorMap(data\_means,  
 # use the constraint from the main map  
 constraints = baseMap.i.12$constraints,  
 col.points = col4Means,  
 cex = 7, # size of the dot (bigger)  
 col.labels = col4Means,axis1 = 1,axis2 = 2,  
 text.cex = 6)  
  
  
MapGroup23 <- PTCA4CATA::createFactorMap(data\_means,  
 # use the constraint from the main map  
 constraints = baseMap.i.23$constraints,  
 col.points = col4Means,  
 cex = 7, # size of the dot (bigger)  
 col.labels = col4Means,axis1 = 2,axis2 = 3,  
 text.cex = 6)  
  
# The map with observations and group means  
I.withMeans12 <- FS\_JL\_12 +  
 MapGroup12$zeMap\_dots + MapGroup12$zeMap\_text  
  
I.withMeans23 <- FS\_JL\_23 +  
 MapGroup23$zeMap\_dots + MapGroup23$zeMap\_text   
  
print(I.withMeans12)

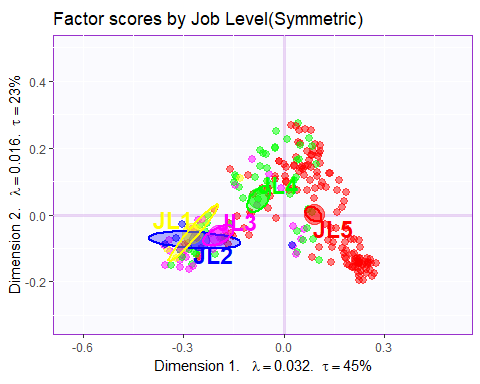


print(I.withMeans23)



### **Confidence Intervals:**

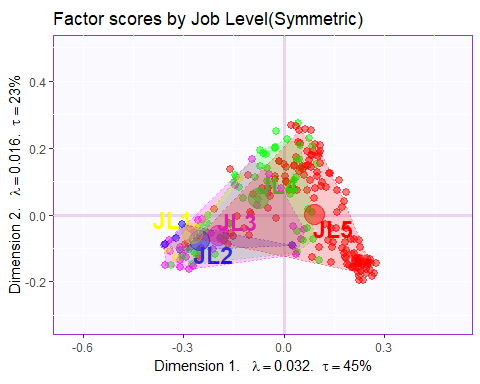
# Confidence Intervals ----  
# Bootstrap for CI:  
BootCube <- PTCA4CATA::Boot4Mean(resMCA.sym$ExPosition.Data$fi,   
 design = DESIGN$rows$JobLevel$color\_observ,  
 niter = 100,  
 suppressProgressBar = TRUE)  
  
GraphElli\_12 <- PTCA4CATA::MakeCIEllipses(BootCube$BootCube[,1:2,],  
 names.of.factors = c("Dimension 1","Dimension 2"),  
 col = rownames(BootCube$BootCube[,c(1,2),1]),  
 p.level = .95)  
  
GraphElli\_23 <- PTCA4CATA::MakeCIEllipses(BootCube$BootCube[,2:3,],  
 names.of.factors = c("Dimension 1","Dimension 2"),  
 col = rownames(BootCube$BootCube[,c(1,2),1]),  
 p.level = .95)  
  
I.JL.withCI\_12 <- I.withMeans12 + GraphElli\_12   
  
print(I.JL.withCI\_12)



**Employee with JobLevel 1 & 2 have clear seperation in the confidence Interval**

### **Tolerance Intervals:**

# Tolerance Intervals ----  
# use function MakeToleranceIntervals from package PTCA4CATA  
GraphTI.Hull <- PTCA4CATA::MakeToleranceIntervals(resMCA.sym$ExPosition.Data$fi,  
 design = as.factor(DESIGN$rows$JobLevel$color\_observ),  
 # line below is needed  
 names.of.factors = c("Dim1","Dim2"), # needed   
 col = rownames(BootCube$BootCube[,c(1,2),1]),  
 line.size = .50,   
 line.type = 3,  
 alpha.ellipse = .2,  
 alpha.line = .4,  
 p.level = .75)  
  
TI.withTIHull <-I.withMeans12 + GraphTI.Hull   
  
#dev.new()  
print(TI.withTIHull)



## Conclusion

* I map showed factor score distribution along Dimension 1 when colored by Job Level
* J map showed few variables as significant after binning as Year (with manager, current role), age. monthly income
* For dimension 2&3 we couldn’t find any pattern for different variables
* Overall, it showed a trend that as experience increases Job Level increased

PLS-C

# PLSC (Partial Least Square Correlation)

PLSC is used when we are trying to find information that is shared between two tables. We have two latent variables after the analysis, for each of the tables involved.

* Latent Variables: Variables obtained as linear combination of original variables.
* Saliences: Each variables are called saliences and contribute according to their weights towards respective latent variables.

### **Splitting the data into 3 tables, 2 for the correlation operation, 3rd for the design**

ibm.year <- ibm.emp.data[10:13]  
ibm.income <- ibm.emp.data[20:28]  
ibm.income <- ibm.income[-c(5)]  
colnames(ibm.income)

## [1] "NumCompaniesWorked" "PercentSalaryHike"   
## [3] "TotalWorkingYears" "TrainingTimesLastYear"   
## [5] "YearsAtCompany" "YearsInCurrentRole"   
## [7] "YearsSinceLastPromotion" "YearsWithCurrManager"

colnames(ibm.year)

## [1] "MonthlyIncome" "DailyRate" "HourlyRate" "MonthlyRate"

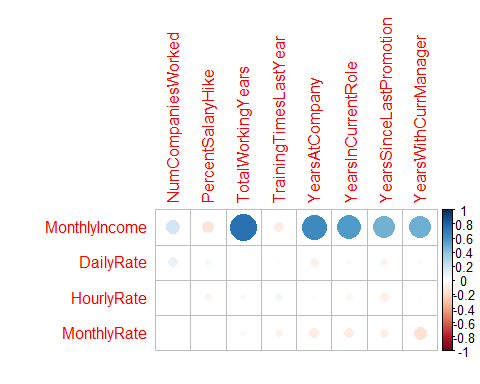
ibm.design <- ibm.emp.data[,18]  
raw.data <- ibm.emp.data[-c(10:13,20:28)]

### **Take small sample out of the tables**

year <- ibm.year#[1:nN,]  
income <- ibm.income#[1:nN,]  
sample.design <- ibm.design#[1:nN]

### **Correlation plot amongst the sample**

## heatmap  
corrplot::corrplot(cor(year,income))



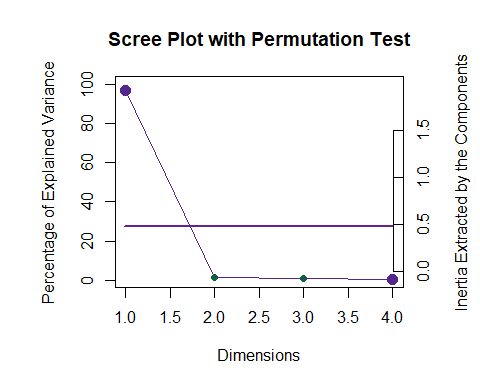
## Running PLSC

resPLSC <- tepPLS(year,income,DESIGN = factor(sample.design),graphs = FALSE)  
  
#resPCA.inf <- epPCA.inference.battery(year,income,DESIGN = sample.design,graphs = FALSE)  
  
Fi <- resPLSC$TExPosition.Data$fi  
Fj <- resPLSC$TExPosition.Data$fj  
constraints.factor <- minmaxHelper(mat1 = Fi, mat2 = Fj)  
  
Eigs <- resPLSC$TExPosition.Data$eigs  
tau <- resPLSC$TExPosition.Data$t  
  
Cj <- resPLSC$TExPosition.Data$cj  
  
label2PLSC <- createxyLabels.gen(x\_axis = 1, y\_axis = 2,  
 lambda = Eigs,tau =tau)  
  
Lx <- resPLSC$TExPosition.Data$lx   
Ly <- resPLSC$TExPosition.Data$ly  
constraints.latent <- minmaxHelper(mat1 = Lx, mat2 = Ly)

### **Permutation Test**

### **Scree Plot of Eigen values**

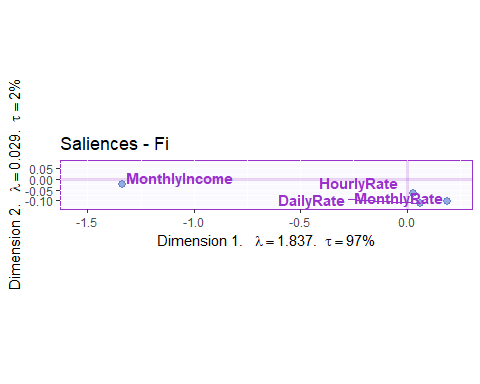
# Eigenv  
EigVal <- resPLSC$TExPosition.Data$eigs   
# How many eigen-value do we have here  
nL <- min(ncol(year),ncol(income))  
  
PlotScree(ev = EigVal,   
 p.ev = resPerm4PLSC$pEigenvalues,  
 title = "Scree Plot with Permutation Test",  
 plotKaiser = TRUE  
 )



a001a.screePlot <- recordPlot()

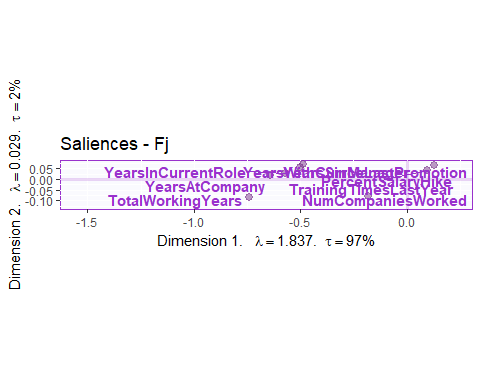
### **Salience - Fi**

salience.i <- createFactorMap(Fi, constraints = constraints.factor,  
 col.points = resPLSC$Plotting.Data$fi.col ,  
 col.labels = 'darkorchid',title = "Saliences - Fi",display.labels = TRUE)   
  
print(salience.i$zeMap + label2PLSC)



### **Salience - Fj**

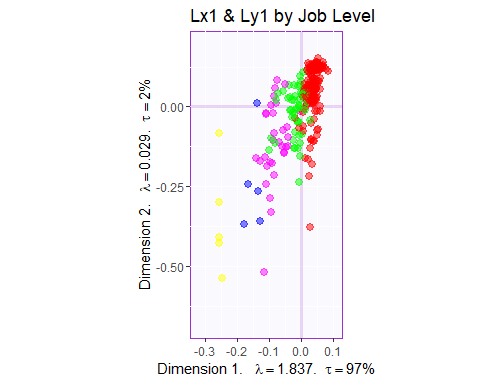
salience.j <- createFactorMap(Fj, constraints = constraints.factor,  
 col.points = resPLSC$Plotting.Data$fj.col ,  
 col.labels = 'darkorchid',title = "Saliences - Fj",display.labels = TRUE)   
  
  
print(salience.j$zeMap + label2PLSC)



* Design Variable: Job Level We will plot Saliences and Factor scores and bootstrap ratios side by side, to show the effect on one table’s variable based on the pattern in the second table

### **LX1 vs Ly1:**

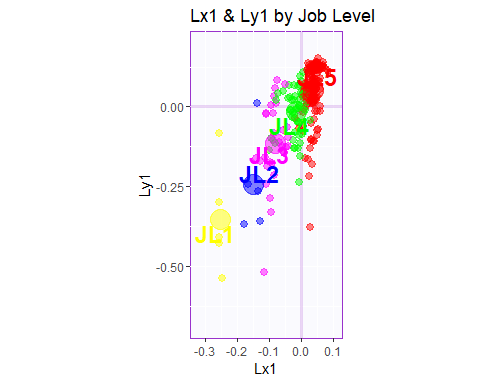
Lx1 <- resPLSC$TExPosition.Data$lx[,1]  
  
Ly1 <- resPLSC$TExPosition.Data$ly[,1]  
  
df <- cbind(Lx1,Ly1)  
  
  
baseMap.i <- createFactorMap(df,# constraints = constraints.sym,  
 col.points = DESIGN$rows$JobLevel$color\_observ,  
 col.labels = 'darkorchid',title = "Lx1 & Ly1 by Job Level",display.labels = FALSE)   
  
print(baseMap.i$zeMap\_background +baseMap.i$zeMap\_dots + label2PLSC)



## Inferences

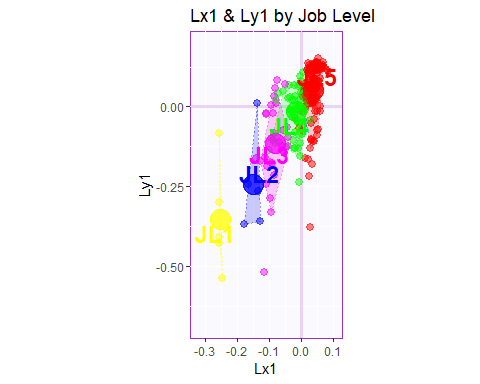
### **Means between Lx1 & Ly1:**

col4data0 <- resPLSC$Plotting.Data$fii.col  
col4Means0 <- unique(col4data0)  
  
  
data\_means <- PTCA4CATA::getMeans(df,ibm.emp.data$JobLevel)  
# a vector of color for the means  
rownames(data\_means) = c("JL5","JL4","JL3","JL2","JL1")  
  
  
MapGroup <- PTCA4CATA::createFactorMap(data\_means,  
 # use the constraint from the main map  
 #constraints = baseMap.i$constraints,  
 col.points = DESIGN$rows$JobLevel$color\_groups,  
 cex = 7, # size of the dot (bigger)  
 display.labels = TRUE,  
 col.labels = DESIGN$rows$JobLevel$color\_groups,  
 text.cex = 6)  
  
  
# The map with observations and group means  
a003.Map.I.withMeans <- baseMap.i$zeMap +MapGroup$zeMap\_dots + MapGroup$zeMap\_text  
  
   
  
print(a003.Map.I.withMeans)



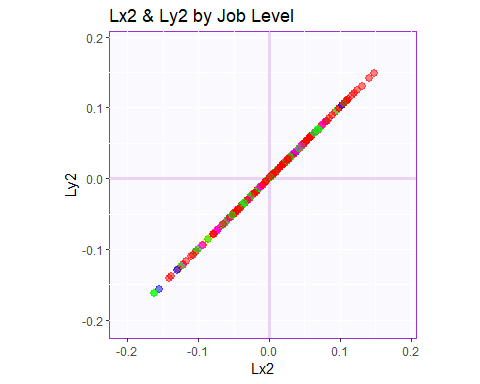
### **Tolerance Intervals:**

# Tolerance Intervals ----  
  
GraphTI.Hull <- PTCA4CATA::MakeToleranceIntervals(df,  
 design = ibm.emp.data$JobLevel,  
 # line below is needed  
 names.of.factors = c("Dim1","Dim2"), # needed   
 #col = rownames(BootCube$BootCube[,c(1,2),1]),  
 col = DESIGN$rows$JobLevel$color\_groups,  
 line.size = .50,   
 line.type = 3,  
 alpha.ellipse = .2,  
 alpha.line = .4,  
 p.level = .75)  
  
  
a005.Map.I.withTIHull <-baseMap.i$zeMap +  
 GraphTI.Hull + MapGroup$zeMap\_dots +  
 MapGroup$zeMap\_text + MapGroup$zeMap\_dots  
  
  
print(a005.Map.I.withTIHull)



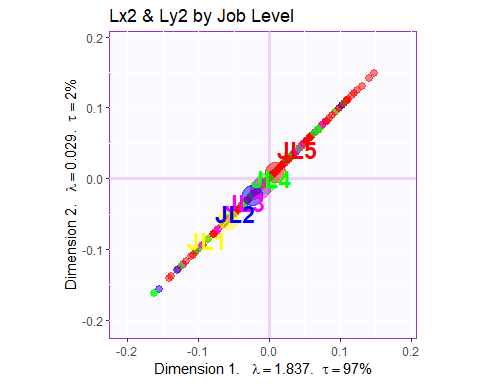
### **Lx2 vs Ly2:**

Lx2 <- resPLSC$TExPosition.Data$lx[,2]  
Ly2 <- resPLSC$TExPosition.Data$lx[,2]  
df1 <- as.data.frame(cbind(Lx2,Ly2))  
#ggplot(NULL, aes(x=Lx2,y=Ly2)) + geom\_point(aes(colour = ibm.emp.data$JobLevel))  
  
  
baseMap1 <- createFactorMap(df1,# constraints = constraints.sym,  
 col.points = DESIGN$rows$JobLevel$color\_observ,  
 col.labels = 'darkorchid',title = "Lx2 & Ly2 by Job Level",display.labels = FALSE)   
  
  
print(baseMap1$zeMap)



### Means between Lx2 & Ly2

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Inferences  
  
#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Mean Map  
  
data\_means <- PTCA4CATA::getMeans(df1,ibm.emp.data$JobLevel)  
rownames(data\_means) = c("JL5","JL4","JL3","JL2","JL1")  
# a vector of color for the means  
col4Data <- DESIGN$rows$JobLevel$color\_groups  
  
MapGroup <- PTCA4CATA::createFactorMap(data\_means,  
 col.points = col4Data,  
 cex = 7, # size of the dot (bigger)  
 col.labels =col4Data,  
 text.cex = 6)  
  
  
# The map with observations and group means  
  
a003.Map.Y.withMeans <- baseMap1$zeMap +MapGroup$zeMap\_dots + MapGroup$zeMap\_text + label2PLSC  
   
print(a003.Map.Y.withMeans)

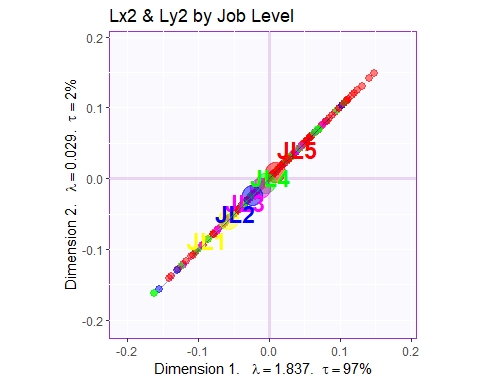


**Lx2 & Ly2 showed perfect correlation**

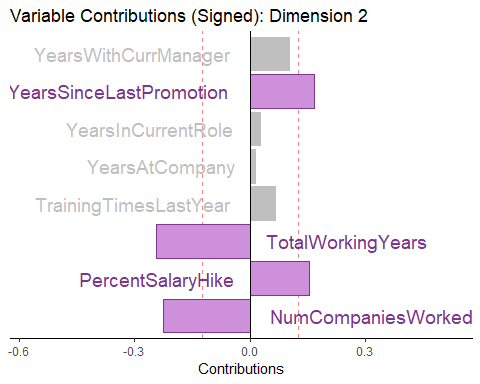
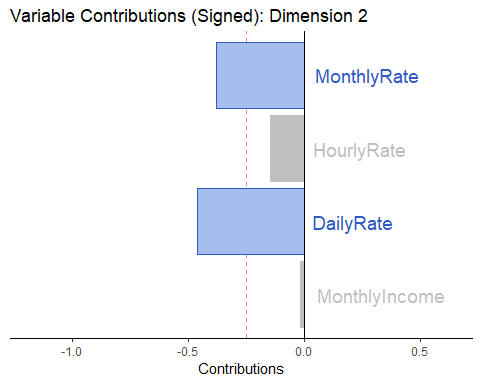
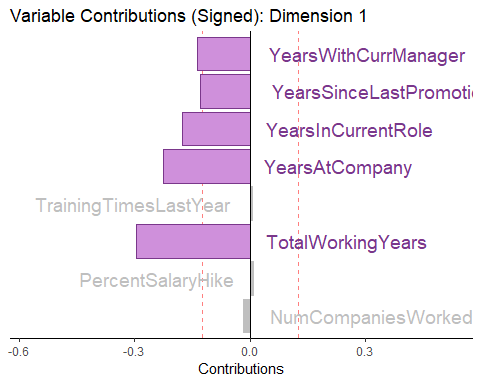
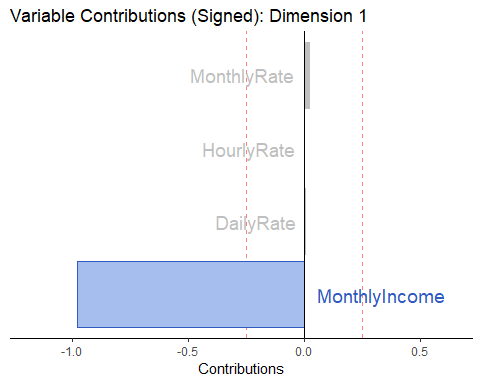
### **Tolerance Intervals:**

# Tolerance Intervals ----  
# use function MakeToleranceIntervals from package PTCA4CATA  
GraphTI.Hull <- PTCA4CATA::MakeToleranceIntervals(df1,  
 design = as.factor(sample.design),  
 # line below is needed  
 names.of.factors = c("Dim1","Dim2"), # needed   
 col = rownames(BootCube$BootCube[,c(1,2),1]),  
 line.size = .50,   
 line.type = 3,  
 alpha.ellipse = .2,  
 alpha.line = .4,  
 p.level = .75)  
#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Create the map:  
a005.Map.Y.withTIHull <-a003.Map.Y.withMeans +  
 GraphTI.Hull

print(a005.Map.Y.withTIHull)



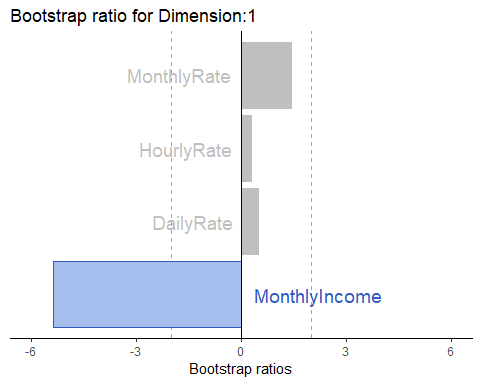
### **Contribution Bars:**



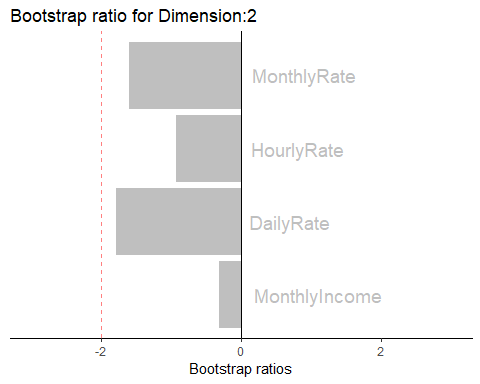
### **Bootstrap Test**

### **Bootstrap Ratios for first table:**

laDim = 1  
ba001.BR1 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.i[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resPLSC$Plotting.Data$fi.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios',horizontal=FALSE  
 #ylim = c(1.2\*min(BR[,laDim]), 1.2\*max(BR[,laDim]))  
)  
#dev.new()  
print(ba001.BR1)



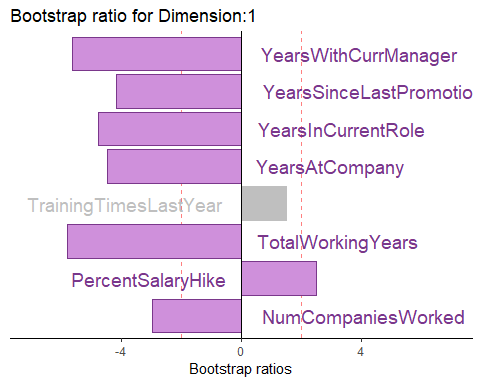
laDim = 2  
ba001.BR2 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.i[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resPLSC$Plotting.Data$fi.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios',horizontal=FALSE  
 #ylim = c(1.2\*min(BR[,laDim]), 1.2\*max(BR[,laDim]))  
)  
print(ba001.BR2)



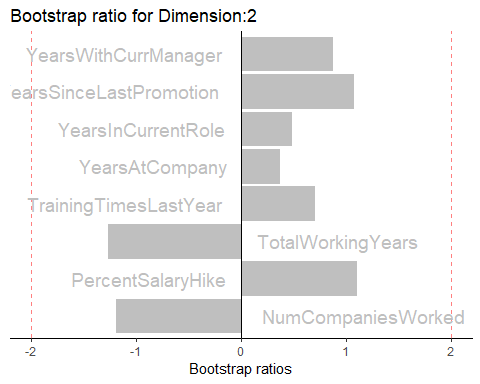
**1st Table: Monthly Income on the -ve side of DImension 1**

### **Bootstrap Ratios for second table:**

laDim = 1  
ba001.BR1 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.j[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resPLSC$Plotting.Data$fj.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios',horizontal=FALSE  
 )  
  
print(ba001.BR1)



laDim = 2  
ba001.BR2 <- PrettyBarPlot2(resBoot4PLSC$bootRatios.j[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resPLSC$Plotting.Data$fj.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios',horizontal=FALSE  
 )  
  
print(ba001.BR2)



**2nd Table: Year realted variables are on the -ve side of DImension 1, and pecent salry hike is on +ve side of Dimension 1**

## Conclusion

* Saliences Fi when considered with factor score plot, showed higher Job Levels coincided with higher monthly income.
* Saliences Fj when considered with factor score plot, showed similar trend as above with all the year related variables.

BADA

# BADA (Barycentric Discriminant Analysis)

Barycentric discriminant analysis (BADA) is a robust version of discriminant analysis that is used to assign, to pre-defined groups (also called categories), observations described by multiple variables. By contrast with traditional discriminant analysis, BADA can be used even when the number of observations is smaller than the number of variables—This makes BADA particularly suited for the analysis of Big Data.

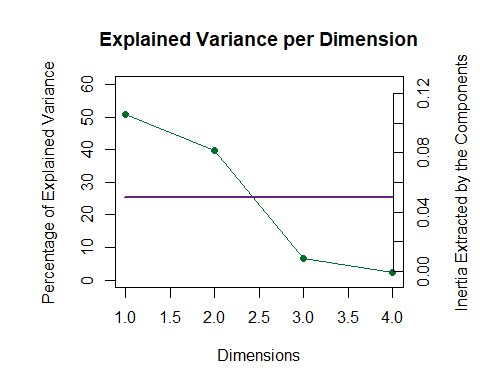
## Running BADA

new.ibm.emp.data1 <- subset(new.ibm.emp.data, select = -c(JobLevel))  
  
  
resBADA <- tepBADA(DATA = new.ibm.emp.data1,  
 scale = 'SS1', center = TRUE,  
 #DESIGN = DESIGN$rows$JobLevel$color\_observ,  
 DESIGN = JL,  
 make\_design\_nominal = TRUE,  
 group.masses = NULL,  
 weights = NULL, graphs = FALSE)  
  
resBADA.inf <- tepBADA.inference.battery(DATA = new.ibm.emp.data1,  
 scale = 'SS1', center = TRUE,  
 #DESIGN = DESIGN$rows$JobLevel$color\_observ ,  
 DESIGN = JL,  
 make\_design\_nominal = TRUE,  
 group.masses = NULL,  
 weights = NULL,  
 graphs = FALSE,  
 test.iters = 100)

## [1] "It is estimated that your iterations will take 0.19 minutes."  
## [1] "R is not in interactive() mode. Resample-based tests will be conducted. Please take note of the progress bar."  
## ===========================================================================

### **Scree Plot**

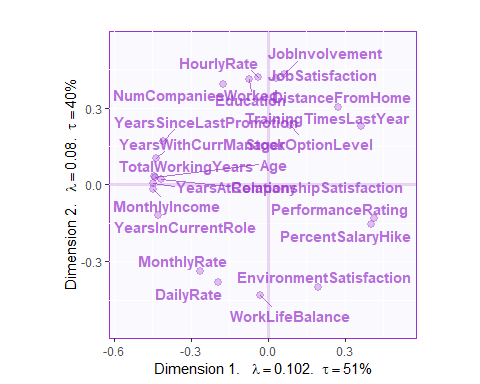
# A look at the scree  
  
PlotScree(ev = resBADA$TExPosition.Data$eigs, title = "Explained Variance per Dimension",plotKaiser = TRUE)



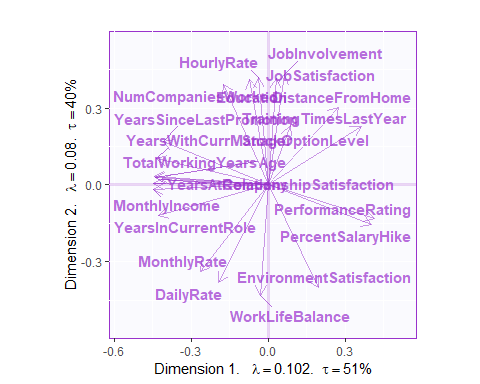
BADAScree <- recordPlot()

### **Loading plots:**

#---------------------------------------------------------------------  
# A graph for the J set  
baseMap.j <- PTCA4CATA::createFactorMap(Fj,  
 col.points = 'darkorchid',  
 alpha.points = .3,  
 alpha.labels = .7,  
 col.labels = 'darkorchid')  
# A graph for the J-set  
aggMap.j <- baseMap.j$zeMap\_background + # background layer  
 baseMap.j$zeMap\_dots + baseMap.j$zeMap\_text # dots & labels  
# We print this Map with the following code  
#dev.new()  
print(aggMap.j + labels4BADA\_12)

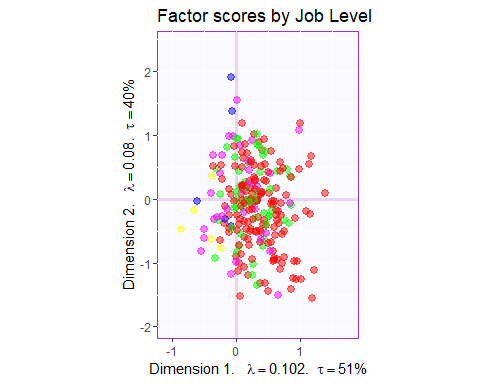


zeLines <- ggplot2::annotate("segment", x = c(0), y = c(0),  
 xend = Fj[,1],  
 yend = Fj[,2],  
 color = 'darkorchid',  
 alpha = .5,  
 arrow = arrow(length = unit(.3, "cm") ) )  
# Create the map by adding background, labels, and arrows:  
aggMap.j.arrows <- baseMap.j$zeMap\_background +  
 zeLines + baseMap.j$zeMap\_text  
  
print(aggMap.j.arrows + labels4BADA\_12)



## I Set map:

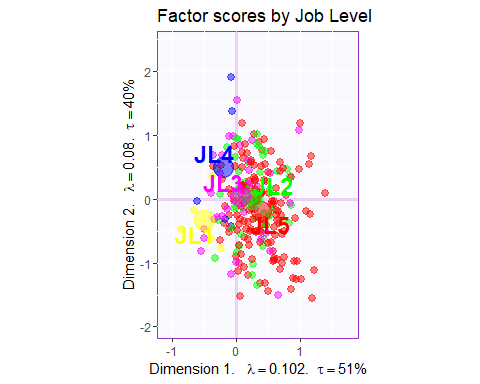
#---------------------------------------------------------------------  
# Graphs for the I set  
  
  
  
baseMap.i <- createFactorMap(Fk,   
 col.points = DESIGN$rows$JobLevel$color\_observ,  
 col.labels = 'darkorchid',title = "Factor scores by Job Level",display.labels = FALSE)   
  
aggMap.i <- baseMap.i$zeMap\_background + baseMap.i$zeMap\_dots  
  
print(baseMap.i$zeMap +labels4BADA\_12)



## Means:

data\_means <- PTCA4CATA::getMeans(Fk, ibm.emp.data$JobLevel)  
rownames(data\_means) = c("JL5","JL2","JL3","JL4","JL1")  
  
# the map  
MapGroup <- PTCA4CATA::createFactorMap(data\_means,  
 col.points = DESIGN$rows$JobLevel$color\_groups,  
 cex = 7, # size of the dot (bigger)  
 display.labels = TRUE,  
 col.labels = DESIGN$rows$JobLevel$color\_groups,  
 text.cex = 6)  
  
  
# The map with observations and group means  
aggMap.i.withMeans <- aggMap.i+  
 MapGroup$zeMap\_dots + MapGroup$zeMap\_text

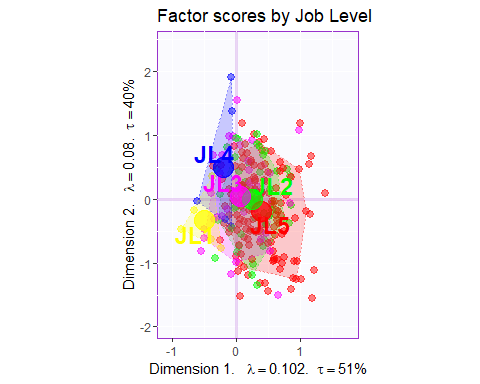
print(aggMap.i.withMeans + labels4BADA\_12)



#---------------------------------------------------------------------

## Tolerance Interval

#---------------------------------------------------------------------  
# Create 75% Tolerance interval polygons  
#  
GraphTI.Hull.90 <- MakeToleranceIntervals(Fk,  
 ibm.emp.data$JobLevel,  
 names.of.factors = c("Dim1","Dim2"),  
 col = DESIGN$rows$JobLevel$color\_groups,  
 line.size = .5, line.type = 3,  
 alpha.ellipse = .2,  
 alpha.line = .4,  
 p.level = .75, # 75% TI  
 type = 'hull' #  
 # use 'hull' for convex hull  
)  
#---------------------------------------------------------------------  
# Create the map  
aggMap.i.withHull <- aggMap.i +  
 GraphTI.Hull.90 + MapGroup$zeMap\_dots +  
 MapGroup$zeMap\_text + MapGroup$zeMap\_dots  
print(aggMap.i.withHull + labels4BADA\_12)



#---------------------------------------------------------------------

### **Confusion Matrix:**

fixedCM <- resBADA.inf$Inference.Data$loo.data$fixed.confuse  
looedCM <- resBADA.inf$Inference.Data$loo.data$loo.confuse  
  
print(fixedCM)

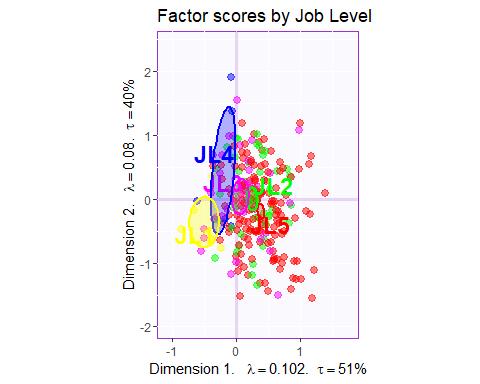
## .2 .1 .5 .3 .4  
## .2 16 15 0 3 0  
## .1 12 74 0 4 1  
## .5 3 7 4 7 2  
## .3 11 37 0 11 0  
## .4 10 10 1 7 2

print(looedCM)

## .2.actual .1.actual .5.actual .3.actual .4.actual  
## .2.predicted 6 20 1 6 3  
## .1.predicted 20 70 1 4 0  
## .5.predicted 3 8 1 8 0  
## .3.predicted 12 36 2 5 2  
## .4.predicted 11 9 0 9 0

### Confidence Intervals

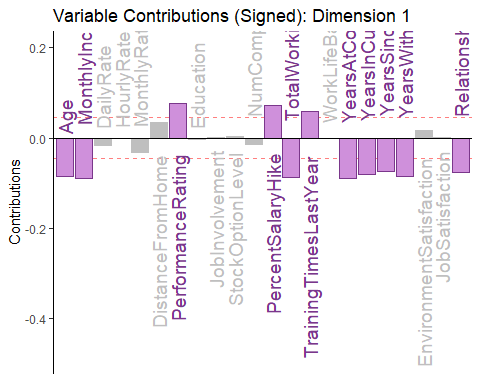
# Confidence Intervals ----  
  
  
# Bootstrap for CI:  
BootCube <- PTCA4CATA::Boot4Mean(Fk,  
 design = ibm.emp.data$JobLevel,  
 niter = 100,  
 suppressProgressBar = TRUE)  
  
  
GraphElli <- PTCA4CATA::MakeCIEllipses(BootCube$BootCube[,1:2,],  
 names.of.factors = c("Dimension 1","Dimension 2"),  
 col = DESIGN$rows$JobLevel$color\_groups,  
   
 p.level = .95  
)  
  
a004.Map.I.withCI <- aggMap.i + MapGroup$zeMap\_text + GraphElli + labels4BADA\_12  
  
print(a004.Map.I.withCI)



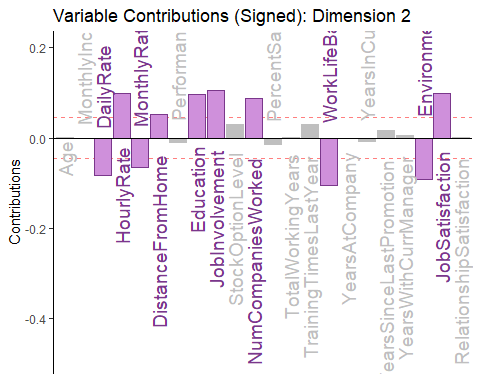
## Inference

## Contribution Plot

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Contribution Plots ----  
  
signed.ctrJ <- resBADA$TExPosition.Data$cj \* sign(resBADA$TExPosition.Data$fj)  
  
b003.ctrJ.bada.1 <- PrettyBarPlot2(signed.ctrJ[,1],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(resBADA$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 1',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
print(b003.ctrJ.bada.1)



b004.ctrJ.bada.2 <- PrettyBarPlot2(signed.ctrJ[,2],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(resBADA$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 2',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
print(b004.ctrJ.bada.2)



Age, Monthly Income,Total Working Years and other Year realted variables lie on the -ve side of dimension 1

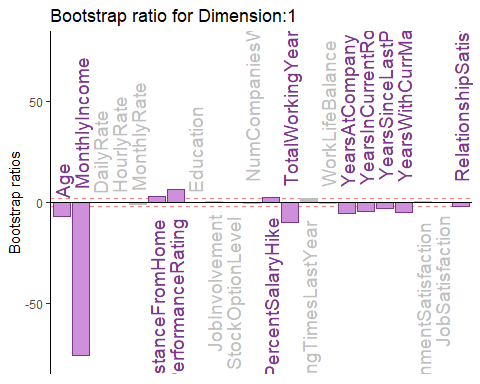
Performance Rating, Percent Salary Hike, Train Times Last Year on +ve side of dimension 1

DailyRate, Monthly Rate,WorkLifeBalance, EnvironmentStifaction on -ve of DImension 2

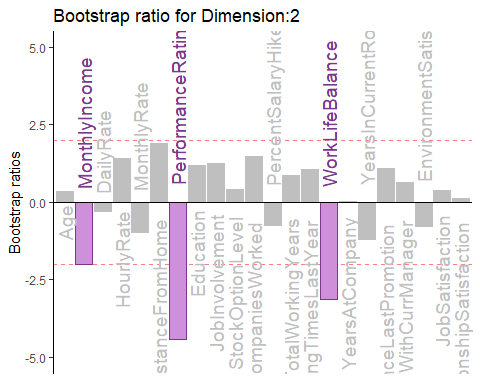
Hourly Rate, DIstFrm Home, Education, Job Invovlement Job Satisfactio on +ve of Dimesnion 2

## Bootstrap Bars

BADA.BR <- resBADA.inf$Inference.Data$boot.data$fj.boot.data$tests$boot.ratios  
laDim = 1  
ba001.bada.BR1 <- PrettyBarPlot2(BADA.BR[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resBADA$Plotting.Data$fj.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios'  
 #ylim = c(1.2\*min(BR[,laDim]), 1.2\*max(BR[,laDim]))  
)  
  
print(ba001.bada.BR1)



#  
laDim = 2  
ba002.bada.BR2 <- PrettyBarPlot2(BADA.BR[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resBADA$Plotting.Data$fj.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios'  
)  
  
print(ba002.bada.BR2)



Age, Monthly Income,Total Working Years and other Year realted variables lie on the -ve side of dimension 1  
Performance Rating, Percent Salary Hike, Train Times Last Year on +ve side of dimension 1  
Monthly Rate,WorkLifeBalance,Perf Rating on -ve of DImension 2

## Conclusion

* BADA was applied and categorized based on Job Level (which) had 5 levels.
* Factor scores were divided amongst the Job levels, with less overlapping.
* Based on which we can predict the levels of supplemental observations.

DiCA

# Discriminant Correspondence Analysis

DICA is an extension on CA and DA (grouping to predefined categories). -The main idea behind DCA is to represent each group by the sum of its observations and to perform a simple CA on the groups by variables matrix.

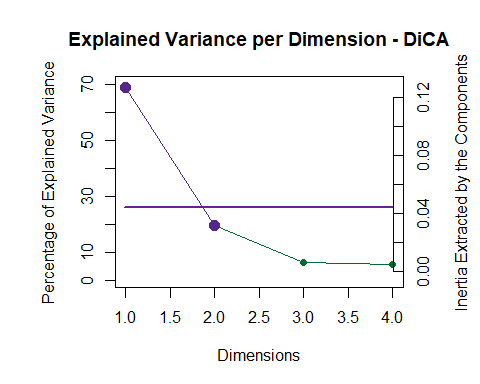
## Running DiCA

resDICA <- tepDICA(DATA = new\_df,  
 DESIGN = JL,  
 make\_design\_nominal = TRUE,  
 make\_data\_nominal = TRUE,  
 group.masses = NULL,  
 weights = NULL, graphs = FALSE)  
  
resDICA.inf <- tepDICA.inference.battery(DATA = new\_df,  
 DESIGN = JL,  
 make\_design\_nominal = TRUE,  
 make\_data\_nominal = TRUE,  
 group.masses = NULL,  
 weights = NULL,  
 graphs = FALSE,  
 test.iters = 100,  
 critical.value = 2)

## [1] "It is estimated that your iterations will take 0.11 minutes."  
## [1] "R is not in interactive() mode. Resample-based tests will be conducted. Please take note of the progress bar."  
## ===========================================================================

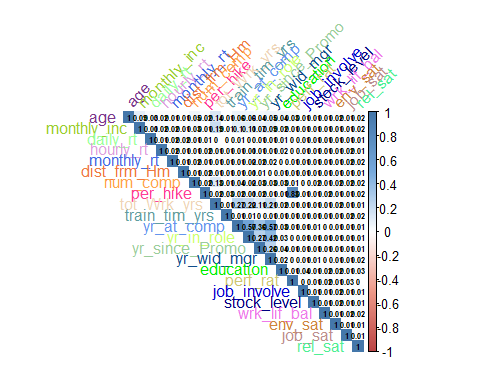
### **DiCA Scree Plot:**

# A look at the scree  
  
PlotScree(ev = resDICA$TExPosition.Data$eigs,  
 p.ev = resDICA.inf$Inference.Data$components$p.vals, max.ev = NULL, alpha = 0.05,  
 col.ns = "#006D2C", col.sig = "#54278F",  
 title = "Explained Variance per Dimension - DiCA",plotKaiser = TRUE)



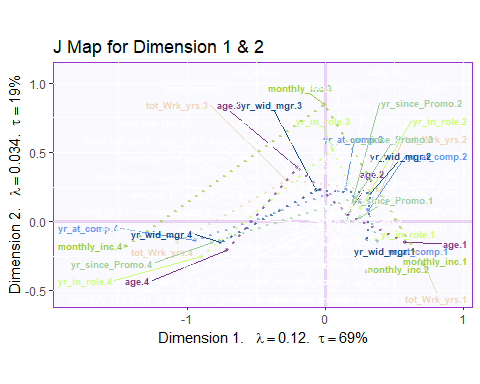
DICAScree <- recordPlot()

### **Heat Map**



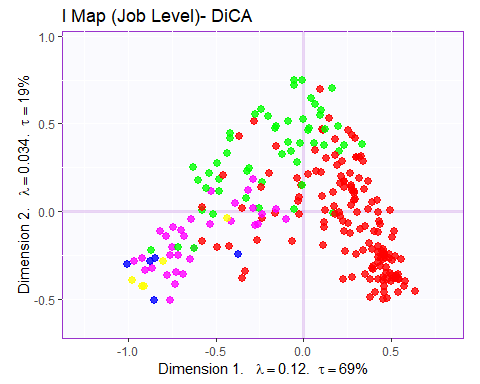
### **Factor Map**

#---------------------------------------------------------------------  
#---------------------------------------------------------------------  
# A graph for the J set  
baseMap.dica.j <- PTCA4CATA::createFactorMap(Fj,  
 col.points = col4Levels.imp$color4Levels,col.labels = col4Levels.imp$color4Levels,  
 alpha.points = .3,display.labels = TRUE,display.points = TRUE,text.cex = 2.5,force=2,cex=1,  
 title = "J Map for Dimension 1 & 2"  
 )  
  
lines4J <- addLines4MCA(Fj, col4Var = col4Levels.imp$color4Variables, size = 1)  
  
# A graph for the J-set  
aggMap.j <- baseMap.dica.j$zeMap\_background + # background layer  
 baseMap.dica.j$zeMap\_dots + baseMap.dica.j$zeMap\_text + lines4J + labels4DICA\_12  
  
  
  
print(aggMap.j)



### **I Set map**

col4DICA <- DESIGN$rows$JobLevel$color\_observ  
  
  
baseMap.dica.i <- PTCA4CATA::createFactorMap(Fk,  
 col.points = DESIGN$rows$JobLevel$color\_observ,  
 alpha.points = .8, title = "I Map (Job Level)- DiCA")  
  
# Plain map with color for the I-set  
aggMap.jl\_12 <- baseMap.dica.i$zeMap\_background + baseMap.dica.i$zeMap\_dots + labels4DICA\_12  
  
print(aggMap.jl\_12)



**Here we are getting a shape similar to Horsehsoe (Goodmans shape)**

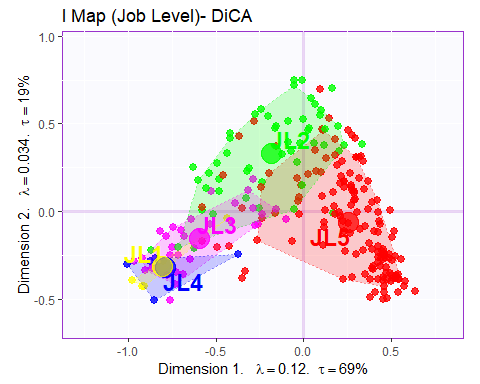
## Inferences

### Means:

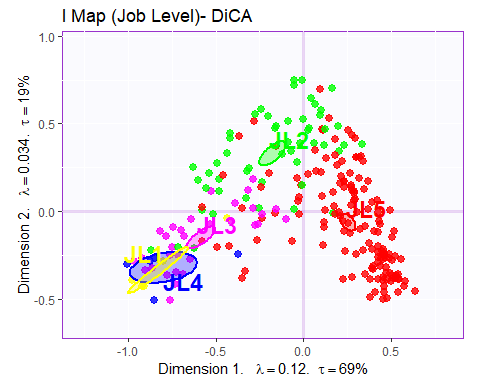
#---------------------------------------------------------------------  
# a vector of color for the means  
col4Means <- unique(col4DICA)  
  
  
data\_means <- PTCA4CATA::getMeans(Fk, ibm.emp.data$JobLevel)  
rownames(data\_means) = c("JL5","JL2","JL3","JL4","JL1")  
  
# the map  
MapGroup <- PTCA4CATA::createFactorMap(data\_means,  
 # use the constraint from the main map  
 #constraints = baseMap.i$constraints,  
 col.points = DESIGN$rows$JobLevel$color\_groups,  
 cex = 7, # size of the dot (bigger)  
 display.labels = TRUE,  
 col.labels = DESIGN$rows$JobLevel$color\_groups,  
 text.cex = 6)  
  
  
# The map with observations and group means  
aggMap.i.withMeans <- aggMap.jl\_12+  
 MapGroup$zeMap\_dots + MapGroup$zeMap\_text  
  
print(aggMap.i.withMeans)  
#---------------------------------------------------------------------

### **Tolerance Interval:**

#---------------------------------------------------------------------  
# Create 75% Tolerance interval polygons  
#  
GraphTI.Hull.90 <- MakeToleranceIntervals(Fk,  
 ibm.emp.data$JobLevel,  
 names.of.factors = c("Dim1","Dim2"),  
 col = DESIGN$rows$JobLevel$color\_groups,  
 line.size = .5, line.type = 3,  
 alpha.ellipse = .2,  
 alpha.line = .4,  
 p.level = .75, # 75% TI  
 type = 'hull' #  
 # use 'hull' for convex hull  
)  
#---------------------------------------------------------------------  
# Create the map  
aggMap.i.withHull <- aggMap.jl\_12 +  
 GraphTI.Hull.90 + MapGroup$zeMap\_dots +  
 MapGroup$zeMap\_text + MapGroup$zeMap\_dots  
  
print(aggMap.i.withHull)

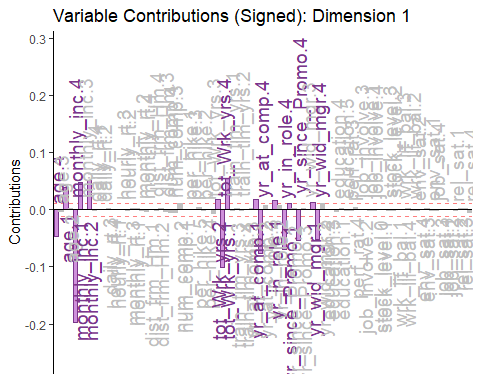


fixedCM <- resDICA.inf$Inference.Data$loo.data$fixed.confuse  
looedCM <- resDICA.inf$Inference.Data$loo.data$loo.confuse  
  
# Bootstrap for CI:  
BootCube <- PTCA4CATA::Boot4Mean(Fk,  
 design = ibm.emp.data$JobLevel,  
 niter = 100,  
 suppressProgressBar = TRUE)  
  
  
GraphElli <- PTCA4CATA::MakeCIEllipses(BootCube$BootCube[,1:2,],  
 names.of.factors = c("Dimension 1","Dimension 2"),  
 col = DESIGN$rows$JobLevel$color\_groups,  
 p.level = .95  
)  
  
  
  
aggMap.i.withCI <- aggMap.jl\_12 + GraphElli + MapGroup$zeMap\_text  
  
print(aggMap.i.withCI)

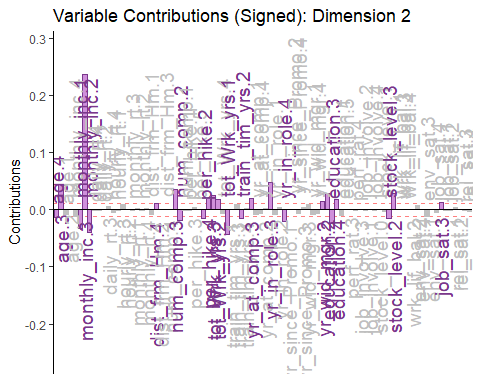


### **Contribution Plot:**

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Contribution Plots ----  
# get the Contributions and make a plot.  
  
signed.ctrJ <- resDICA$TExPosition.Data$cj \* sign(resDICA$TExPosition.Data$fj)  
  
b003.ctrJ.s.1 <- PrettyBarPlot2(signed.ctrJ[,1],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(resDICA$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 1',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
print(b003.ctrJ.s.1)



b004.ctrJ.s.2 <- PrettyBarPlot2(signed.ctrJ[,2],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(resDICA$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 2',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
  
print(b004.ctrJ.s.2)



For Dimension 1: -Age3,MonthlyInc3,Tot\_Wrk\_Yr\_(2,3),numofComp2,yr\_at\_comp3,Job\_sat3 all these lie on the +ve half of the dimension 1. -MonthlyInc(1,2),distFrmHm\_4,numofComp3,per\_hike2 all these lie on the -ve half of the dimension 1.

For Dimension 2: Also shows a similar trend for almost all the same variables

## Conclusion

* Dica also shows a similar pattern as for MCA when colored by Job Level

MFA

# Multiple Factor Analysis

MFA is used to handle multiple tables, that have different variables measuring the same observations, or same set of variables on different set of observations. - First, each table is normalized by 1st singular value (obtained by doing PCA on the same table) - All these normalized tables are merged together and then again a PCA is done on this, which gives us factor score and loading on these variables.

### **Removing Factor variables**

new.ibm.emp.data <- ibm.emp.data[sapply(ibm.emp.data, function(x) !is.factor(x))]

### **Splitting the data into 3 tables, 2 for the correaltion operation, 3rd for the deisgn**

new.ibm.emp.data1 <- subset(new.ibm.emp.data, select = -c(JobLevel))  
  
dim(new.ibm.emp.data1)

## [1] 237 22

colnames(new.ibm.emp.data1)

## [1] "Age" "MonthlyIncome"   
## [3] "DailyRate" "HourlyRate"   
## [5] "MonthlyRate" "DistanceFromHome"   
## [7] "PerformanceRating" "Education"   
## [9] "JobInvolvement" "StockOptionLevel"   
## [11] "NumCompaniesWorked" "PercentSalaryHike"   
## [13] "TotalWorkingYears" "TrainingTimesLastYear"   
## [15] "WorkLifeBalance" "YearsAtCompany"   
## [17] "YearsInCurrentRole" "YearsSinceLastPromotion"   
## [19] "YearsWithCurrManager" "EnvironmentSatisfaction"   
## [21] "JobSatisfaction" "RelationshipSatisfaction"

column.design <- c(1,1,1,1,1,1,2,2,2,2,2,2,3,3,2,3,3,3,3,2,2,2)  
  
column.design = t(as.matrix((column.design)))  
colnames(column.design) <- colnames((new.ibm.emp.data1))

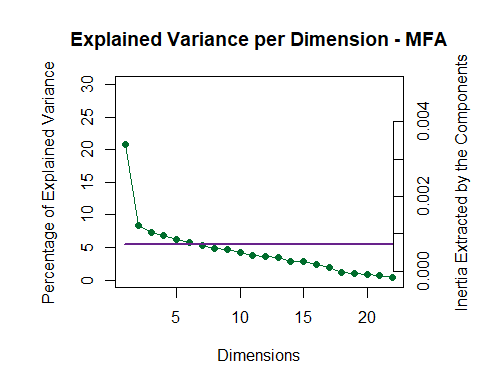
## Running MFA

#new.ibm.emp.data1 <- subset(new.ibm.emp.data, select = -c(JobLevel))  
res.mfa <- mpMFA(new.ibm.emp.data1,as.matrix(column.design),make.columndesign.nominal = TRUE,graphs = FALSE)

## [1] "Preprocessed the Rows of the data matrix using: None"  
## [1] "Preprocessed the Columns of the data matrix using: Center\_1Norm"  
## [1] "Preprocessed the Tables of the data matrix using: MFA\_Normalization"  
## [1] "Preprocessing Completed"  
## [1] "Optimizing using: None"  
## [1] "Processing Complete"

### **Scree Plot**

# A look at the scree  
  
PlotScree(ev = res.mfa$mexPosition.Data$Table$eigs,  
 p.ev = NULL, max.ev = NULL, alpha = 0.05,  
 col.ns = "#006D2C", col.sig = "#54278F",  
 title = "Explained Variance per Dimension - MFA",plotKaiser = TRUE)

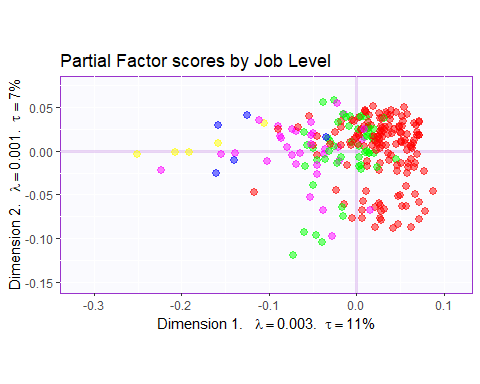


MFAScree <- recordPlot()

**Scree plot gives us 5 components that are above the Kaiser Line**

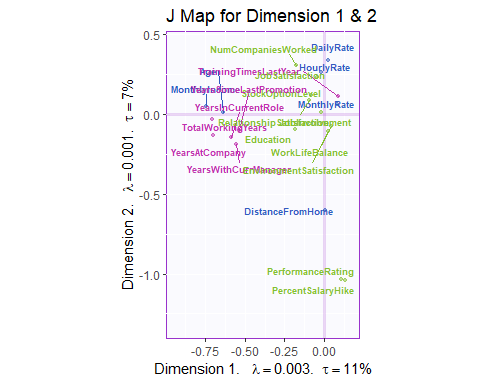
### **Factor Maps**

baseMap.i <- createFactorMap(res.mfa$mexPosition.Data$Table$fi,   
 col.points = DESIGN$rows$JobLevel$color\_observ,  
 col.labels = 'darkorchid',title = "Partial Factor scores by Job Level",display.labels = FALSE,display.points = TRUE)   
  
x\_y\_labels <- createxyLabels.gen(  
1,2, lambda = res.mfa$mexPosition.Data$Table$eigs,  
tau = res.mfa$mexPosition.Data$Table$t  
)  
#dev.new()  
  
print(baseMap.i$zeMap + x\_y\_labels)



### **Loadings**

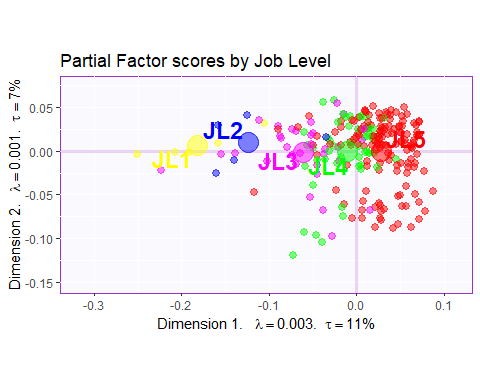
baseMap.j.12 <- createFactorMap(Fj, #constraints = constraints.sym,  
 col.points = res.mfa$Plotting.Data$fj.col,col.labels = res.mfa$Plotting.Data$fj.col,  
 display.labels = TRUE,display.points = TRUE,text.cex = 2.5,force=2,cex=1,  
 title = "J Map for Dimension 1 & 2")  
  
  
#lines4J <- addLines4MCA(Fj, col4Var = col4Levels.imp$color4Variables, size = 1)  
  
Loadings\_12 <- baseMap.j.12$zeMap+ x\_y\_labels  
  
print(Loadings\_12)



## Inferences

### Means

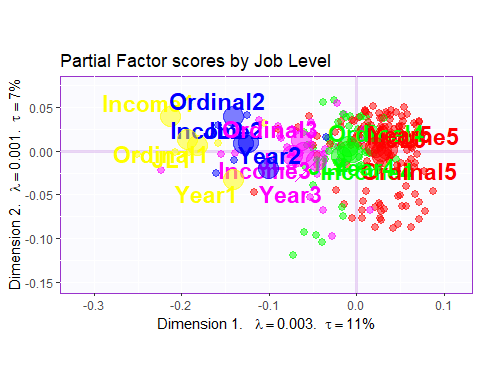
#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Inferences  
  
  
data\_means <- PTCA4CATA::getMeans(res.mfa$mexPosition.Data$Table$fi, ibm.emp.data$JobLevel)  
rownames(data\_means) = c("JL5","JL4","JL3","JL2","JL1")  
  
# the map  
MapGroup <- PTCA4CATA::createFactorMap(data\_means,   
 col.points = DESIGN$rows$JobLevel$color\_groups,  
 cex = 7, # size of the dot (bigger)  
 display.labels = TRUE,  
 col.labels = DESIGN$rows$JobLevel$color\_groups,  
 text.cex = 6)  
  
  
# The map with observations and group means  
a003.Map.I.withMeans <- baseMap.i$zeMap +  
 MapGroup$zeMap\_dots + MapGroup$zeMap\_text + x\_y\_labels  
  
print(a003.Map.I.withMeans)



### **Partial Factor Scores**

Here we calculate the partial factor scores with respect to the 3 tables that we created, to check the inertiaof each.

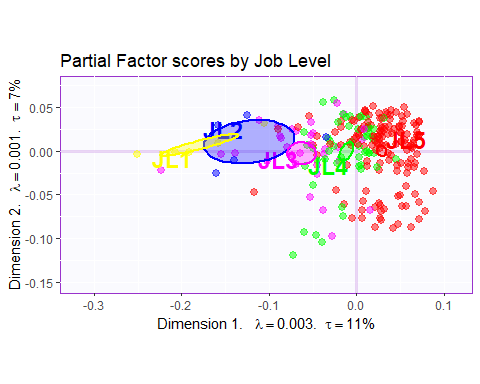
data\_means1 <- PTCA4CATA::getMeans(res.mfa$mexPosition.Data$Table$partial.fi[1:237,], ibm.emp.data$JobLevel)  
data\_means2 <- PTCA4CATA::getMeans(res.mfa$mexPosition.Data$Table$partial.fi[238:474,], ibm.emp.data$JobLevel)  
data\_means3 <- PTCA4CATA::getMeans(res.mfa$mexPosition.Data$Table$partial.fi[475:711,], ibm.emp.data$JobLevel)  
rownames(data\_means1) = c("Income5","Income4","Income3","Income2","Income1")  
rownames(data\_means2) = c("Ordinal5","Ordinal4","Ordinal3","Ordinal2","Ordinal1")  
rownames(data\_means3) = c("Year5","Year4","Year3","Year2","Year1")  
  
# the map  
MapGroup1 <- PTCA4CATA::createFactorMap(data\_means1,   
 col.points = DESIGN$rows$JobLevel$color\_groups,  
 cex = 7, # size of the dot (bigger)  
 display.labels = TRUE,  
 col.labels = DESIGN$rows$JobLevel$color\_groups,  
 text.cex = 6)  
  
MapGroup2 <- PTCA4CATA::createFactorMap(data\_means2,  
 col.points = DESIGN$rows$JobLevel$color\_groups,  
 cex = 7, # size of the dot (bigger)  
 display.labels = TRUE,  
 col.labels = DESIGN$rows$JobLevel$color\_groups,  
 text.cex = 6)  
  
MapGroup3 <- PTCA4CATA::createFactorMap(data\_means3,   
 col.points = DESIGN$rows$JobLevel$color\_groups,  
 cex = 7, # size of the dot (bigger)  
 display.labels = TRUE,  
 col.labels = DESIGN$rows$JobLevel$color\_groups,  
 text.cex = 6)  
  
partial\_plot <- a003.Map.I.withMeans+MapGroup1$zeMap\_dots + MapGroup1$zeMap\_text+MapGroup2$zeMap\_dots + MapGroup2$zeMap\_text+MapGroup3$zeMap\_dots + MapGroup3$zeMap\_text  
print(partial\_plot + x\_y\_labels)



### **Confidence Intervals**

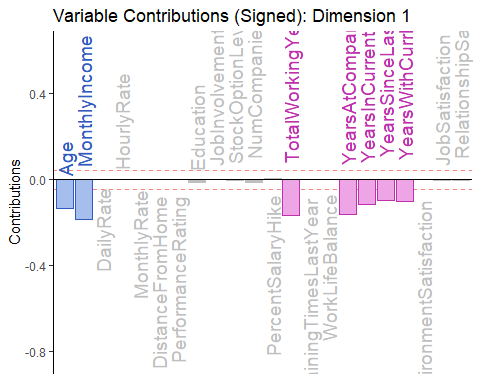
Confidence interval shows no overlap amongst the region of JOb Level except for between Job Level 1 & 2.

# Confidence Intervals ----  
# Bootstrap for CI:  
BootCube <- PTCA4CATA::Boot4Mean(res.mfa$mexPosition.Data$Table$fi,   
 design = DESIGN$rows$JobLevel$color\_observ,  
 niter = 100,  
 suppressProgressBar = TRUE)  
  
  
GraphElli <- PTCA4CATA::MakeCIEllipses(BootCube$BootCube[,1:2,],  
 names.of.factors = c("Dimension 1","Dimension 2"),  
 col = rownames(BootCube$BootCube[,c(1,2),1]),  
 p.level = .95  
)  
  
a004.Map.I.withCI <- baseMap.i$zeMap + MapGroup$zeMap\_text + GraphElli  
  
#dev.new()  
print(a004.Map.I.withCI + x\_y\_labels)

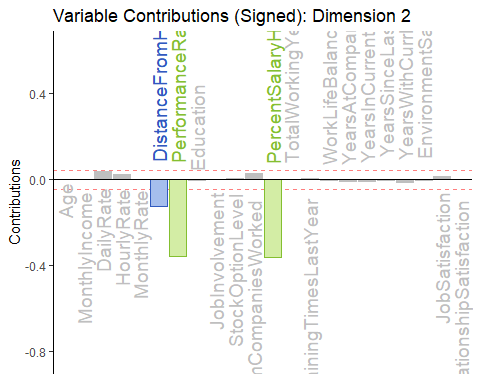


### **Contribution Plot**

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Contribution Plots ----  
  
signed.ctrJ <- res.mfa$mexPosition.Data$Table$cj \* sign(res.mfa$mexPosition.Data$Table$Q)  
  
b003.ctrJ.s.1 <- PrettyBarPlot2(signed.ctrJ[,1],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(res.mfa$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 1',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
  
print(b003.ctrJ.s.1)



b004.ctrJ.s.2 <- PrettyBarPlot2(signed.ctrJ[,2],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(res.mfa$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 2',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
  
print(b004.ctrJ.s.2)



Dimension1: Age,Monthly Income,TotalWorkingYears,YearsAtCompany,YearsInCurrentRole are all on the -ve side of Dimension1 ## Dimension2: WorkLifeBalance & TrainTimeLastYear are on the -ve side of Dimension2

WorkLifeBalance and TrainingTimeLastYear are coming at extreme ends of dimension 2, hence are -vely correlated.

## Conclusion

* In Partial Factor score map, Year table had more inertia as compared to the other two for higher Job Levels
* SO as Job Level increases, experience increases, and it becomes an important factor compared to other variables

**CONCLUSION**

The IBM dataset consisted of details of IBM employees who faced the attrition, in fact the attrition column in the dataset could not be of much use as all the data was having attrition as yes.

So, the techniques could not be used to derive a certain outcome that was expected of it, rather we applied these techniques to check for the general trend inside the data itself.

**PCA:** Dimension 1 showed difference amongst Factor score when colored by Job Level.

* + Dimension 2 (+ve) : Job Satisfaction, Num of Companies Worked.
  + Dimension 2 (-ve) : Percent Salary Hike, Performance Rating
  + Dimension 1 (-ve) : Year (at Company, Current manager, Current Role)

**MCA:** Also showed difference for Job Levels, and on plotting the loadings after binning, Years (at company, with manager) showed similar pattern as Job Level. (Job Level Increased as years passed on)

**Dica:** Similar plots and results as for the MCA

**PLSC:** Saliences Fi() when considered with factor score plot, showed higher Job Levels coincided with higher monthly income.

Saliences Fj() when considered with factor score plot, showed similar trend as above with all the year related variables.

**MFA:** As Job Level increased, the table with year variables had more inertia as compared to the Ordinal and the Income ones.

**PCA showed a pattern with Job Level and the same pattern was seen through till MFA, and MFA showed some new insights on employees on higher positions.**

## Data set: Bird Brain

The Bird Brain data set contains data about 28 different bird’s species, and they are measured using 11 variables, one being categorical and remaining 10 quantitative.

One categorical variable groups these birds into different levels.

(**ROWS**)**: 28**

Number of variables measured for these birds (**COLUMNS**): **10**

The variables represent different brain regions, and we are counting number of neurons present in those areas, for all the birds under consideration.

A head command shows the following snap:

![A screenshot of a cell phone

Description generated with very high confidence](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAkACQAAD/4RDcRXhpZgAATU0AKgAAAAgABAE7AAIAAAAGAAAISodpAAQAAAABAAAIUJydAAEAAAAMAAAQyOocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGFqYXljAAAFkAMAAgAAABQAABCekAQAAgAAABQAABCykpEAAgAAAAMyMQAAkpIAAgAAAAMyMQAA6hwABwAACAwAAAiSAAAAABzqAAAACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMjAxODoxMToyMSAyMTozOToxNgAyMDE4OjExOjIxIDIxOjM5OjE2AAAAYQBqAGEAeQBjAAAA/+ELGGh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8APD94cGFja2V0IGJlZ2luPSfvu78nIGlkPSdXNU0wTXBDZWhpSHpyZVN6TlRjemtjOWQnPz4NCjx4OnhtcG1ldGEgeG1sbnM6eD0iYWRvYmU6bnM6bWV0YS8iPjxyZGY6UkRGIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iLz48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOnhtcD0iaHR0cDovL25zLmFkb2JlLmNvbS94YXAvMS4wLyI+PHhtcDpDcmVhdGVEYXRlPjIwMTgtMTEtMjFUMjE6Mzk6MTYuMjEzPC94bXA6Q3JlYXRlRGF0ZT48L3JkZjpEZXNjcmlwdGlvbj48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOmRjPSJodHRwOi8vcHVybC5vcmcvZGMvZWxlbWVudHMvMS4xLyI+PGRjOmNyZWF0b3I+PHJkZjpTZXEgeG1sbnM6cmRmPSJodHRwOi8vd3d3LnczLm9yZy8xOTk5LzAyLzIyLXJkZi1zeW50YXgtbnMjIj48cmRmOmxpPmFqYXljPC9yZGY6bGk+PC9yZGY6U2VxPg0KCQkJPC9kYzpjcmVhdG9yPjwvcmRmOkRlc2NyaXB0aW9uPjwvcmRmOlJERj48L3g6eG1wbWV0YT4NCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgPD94cGFja2V0IGVuZD0ndyc/Pv/bAEMABwUFBgUEBwYFBggHBwgKEQsKCQkKFQ8QDBEYFRoZGBUYFxseJyEbHSUdFxgiLiIlKCkrLCsaIC8zLyoyJyorKv/bAEMBBwgICgkKFAsLFCocGBwqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKv/AABEIAK8D7wMBIgACEQEDEQH/xAAfAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJCgv/xAC1EAACAQMDAgQDBQUEBAAAAX0BAgMABBEFEiExQQYTUWEHInEUMoGRoQgjQrHBFVLR8CQzYnKCCQoWFxgZGiUmJygpKjQ1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6wsPExcbHyMnK0tPU1dbX2Nna4eLj5OXm5+jp6vHy8/T19vf4+fr/xAAfAQADAQEBAQEBAQEBAAAAAAAAAQIDBAUGBwgJCgv/xAC1EQACAQIEBAMEBwUEBAABAncAAQIDEQQFITEGEkFRB2FxEyIygQgUQpGhscEJIzNS8BVictEKFiQ04SXxFxgZGiYnKCkqNTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqCg4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2dri4+Tl5ufo6ery8/T19vf4+fr/2gAMAwEAAhEDEQA/APo37NB/zxj/AO+BR9mg/wCeMf8A3wKkooAj+zQf88Y/++BR9mg/54x/98CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/wA8Y/8AvgUfZoP+eMf/AHwKkooAj+zQf88Y/wDvgUfZoP8AnjH/AN8CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf8APGP/AL4FH2aD/njH/wB8CpKKAI/s0H/PGP8A74FH2aD/AJ4x/wDfAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf88Y/++BR9mg/54x/98CpKKAI/s0H/ADxj/wC+BR9mg/54x/8AfAqSigCP7NB/zxj/AO+BR9mg/wCeMf8A3wKkooAj+zQf88Y/++BR9mg/54x/98CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/wA8Y/8AvgUfZoP+eMf/AHwKkooAj+zQf88Y/wDvgUfZoP8AnjH/AN8CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf8APGP/AL4FH2aD/njH/wB8CpKKAI/s0H/PGP8A74FH2aD/AJ4x/wDfAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf88Y/++BR9mg/54x/98CpKKAI/s0H/ADxj/wC+BR9mg/54x/8AfAqSigCP7NB/zxj/AO+BR9mg/wCeMf8A3wKkooAj+zQf88Y/++BR9mg/54x/98CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/wA8Y/8AvgUfZoP+eMf/AHwKkooAj+zQf88Y/wDvgUfZoP8AnjH/AN8CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf8APGP/AL4FH2aD/njH/wB8CpKKAI/s0H/PGP8A74FH2aD/AJ4x/wDfAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf88Y/++BR9mg/54x/98CpKKAI/s0H/ADxj/wC+BR9mg/54x/8AfAqSigCP7NB/zxj/AO+BR9mg/wCeMf8A3wKkooAj+zQf88Y/++BR9mg/54x/98CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/wA8Y/8AvgUfZoP+eMf/AHwKkooAj+zQf88Y/wDvgUfZoP8AnjH/AN8CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf8APGP/AL4FH2aD/njH/wB8CpKKAI/s0H/PGP8A74FH2aD/AJ4x/wDfAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf88Y/++BR9mg/54x/98CpKKAI/s0H/ADxj/wC+BR9mg/54x/8AfAqSigCP7NB/zxj/AO+BR9mg/wCeMf8A3wKkooAj+zQf88Y/++BR9mg/54x/98CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/wA8Y/8AvgUfZoP+eMf/AHwKkooAj+zQf88Y/wDvgUfZoP8AnjH/AN8CpKKAI/s0H/PGP/vgUfZoP+eMf/fAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf8APGP/AL4FH2aD/njH/wB8CpKKAI/s0H/PGP8A74FH2aD/AJ4x/wDfAqSigCP7NB/zxj/74FH2aD/njH/3wKkooAj+zQf88Y/++BSGC3UZaKMD1KipaoXcjPdrEmTtwSAOnv8ArQBaWG3b7scR+iil+zQf88Y/++BUccYjuBtzypz+YqxQBH9mg/54x/8AfAo+zQf88Y/++BUGpO62eI3dWZguIly7eynsfftyfeo9K3pHJDJvTyyAIXJYx+wf+JfQ9e3sAC39mg/54x/98Cj7NB/zxj/74FSVma1NdQxQHTt7XRkwkW3KSDuHP8Ixzn1x16EAv/ZoP+eMf/fAo+zQf88Y/wDvgVV0aSWXTUe5eVpyx80SJtKNnlQPQdB147nrV6gAooooAKKKKACiiuM+Kem22o+CZSzSx6lbypJpMtuAZlvM4iCZIySTggkDBOeOaAOzorhfhXeLq2h3ep6kpHiSS5aHWVkQK8MycCIDJwirjbzgg56k13VABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFcL8UbiSOz0K3nme30a81aKDVZUcoBAQ2FZgRhWfYp9Qcd6pTadp3hj4p6HZeD7aDThd2lw+o2NlGIoWiUDZMyKNu4P8obqQSOccAHo9V1v7N9QewS7ga8jjEr2wkBkVCSAxXqASCM9OK8ss/FviO3+Dp8U3ervdX11Itsm61iEVqGuDH5u1VBJCnJydvyjgc5o3uo6j4T+IHjK/tr2XX9Q0/wALwyo9xHGHZhJIwDLEqrgZzwAcY69SAe00V5TZ+JPF9r4Z1+9vJbv7OujfbtOu7/7CZfNCndsW3ZlaLlSCwyM4JOa7rwgb+bwxZXmralJqFzeQRTuzRRxrGWQZVAijC5yfmLHnrjAoA3KKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiorq3S7tJraYZjmRo3HqCMGgCWoJb60gu4LWe6hjuLjd5MLyAPLtGTtU8nA64rwW01zUNJtNM1m5kuhaeA7g6TewohYzB2aMsepOE8hhj+9+XQz6LeW/iLwBbLdyWN9d/b7m5uERXkSSWPe+3cCuQWwMgjjpQB7BVe8v7PTo0k1C7gtUkkWJGnkCBnY4VQT1JPAHU15XpvjfxDqXhTwxBJftBc6prVxplzqqQx70SJpAGClTGHfYByuOTxWf4m1rUdR0GbTtSuDfjR/GNhaQ6gUVGuV8xG+YKAu5SxU7QBx0FAHtlFeUDxb4xvvGWpf2dBdNa6ZrEdnNbFrKO1S2O3c8jO/n+YQxZdvyn5QAeas/FaS01u38NQWMenayv8AwkEcEltPMDC7hXzHIQr49xtOPSgD06ivENN1O78O2cuk+HrpdKum8WRWt5aQ20b21osqBjFB2K453YU5Jyq9K2PiP4t1Lw1aXlvoes61dano+mreXBS3sTBgsQr3BdVJ3YIxDgjHTpQB6h9utP7Q+w/aoftnl+b9n8weZszjdt64zxnpU9edahrN3H8Q7pohAjL4Ve6Vxbxl1k3kj5yN23/ZJx7Vm+CfFXiO51rwWmrau+oReIdDlvLiJ7eKNYpUEZDIUUHkMcgkjPQDpQB6vRRRQAUUUUAFFFFABRRXnPxui+0eCtOh+zw3Xma3Yr5FwcRy5lA2scH5T0PB47HpQB6NVe8v7PTo0k1C7gtUkkWJGnkCBnY4VQT1JPAHU15vb6VJ8P8Aw9qUs8zaPNrWqRiz03w4kciwMQFWOLz0EeW25Ziijr6ZOBq2uanrnw7tG1qSaS4s/GNtahrhYllKpcLjf5X7vcM4JXjigD2GbXNJtxdG41OziFmVW533CL5Bb7ofJ+XORjPWrwORkcivFPEd3JYv8TLiBYWdbrT8CeBJk5VByrgqfxFa3ijxtrOk33j6ODUVt4tI0K3ubANFHiKd1k55HzEkLwcj0FAHqtFeTa94m8TPfNBYa49gkfhT+1GMdrC7NOD1y6kAHoRjp0wea0vBXiLXLzxdZWer6m17Ff8Ah+HUjGYI41hlLBSE2qGwc9GZuemOlAHo9FFFABRRRQAUUUUAFFFFABRRRQAUVx3xc/5I/wCJ/wDsHyfyrmdU0LR/Dkvgq78KafZ6XrN1dwxMtjEITdQFMzCRUA3qB82Wzg4NAHp9tf2d7JcR2d3BcPbSeVOsUgYxPgHawH3Tgg4PPNWK8Zu/F2raPD4vfTo445D4qisfPtra3R4o3SPL5bajPzgNKSMkZyBiruqeJvGmleE8Xks1hejxDa2cE92trLPJbSuv+tSEmNW5YfLgkAHigD1mivJLrxv4h0TSvGdu14dQm0jVLa0t765hjDQRTLHmRxGqqQm8noPetrxhrOseEvDFrFaavdarcXWoQ2z3pjtVngST0yEhySMKXAHzDOaAPQagvL60062NxqF1DawKQDLPIEUEnAGTxyTivL7/AMTeL9N0PTodRun026l8RQ2Pn3a2skz20nI81YiY1c5x8uOgOOecvxnqeqzeFfGGi3eqf2oml6hYpb300UYcF2RjHIIwikqcHgLwQD60Ae11DeXltp9nLd39xFa20Kl5ZpnCIijuWPAH1rzoeKb3RL3xlZa7rt3La6XBbzW94ltD58RlU/IihAjfNjbuB68k1z2sa9rGpfDX4kadrUl840+0Xyf7RS2FwqvFuIf7P+7PPIxzzzQB7WjrJGrxsHRgCrKcgj1Bpa8wl1vWbrWbnSbXWn0G10nw/BfxyxwxP9pdgcl/MRvkXYAQu089elVLfxf4l8Rat4chj1CTQ11Pw1LqVxFBbxuyzKyYKmVGwOehBGD64IAPWqK5r4da5eeJfhzoesamVa7u7RXmZF2hm6E4HTOM10tABRRRQAUUUUAFFFFABRRUN40yWM7Wq7p1jYxrjq2OP1oAmorzP4b6L4b1HwXZeI9WtrOfXnYvqOpz4+1Q3WcOhl+8m0/KFBGBgYxVa18WeLtR8X3r2cF19j0/WRY3Fu7WSWsdvwNzFn8/zTuDDHynIABoA9VqK5lggt5Jrt444YlLvJKQFQDqSTwB7155pXiLX57vx1fSX0t1FoF7NFZackEYWQCBXCswXeeTxgg8nJPAGJZ694n1Xwbqk+rfap9L1Dw9Ncxz3jWQxNs5EIt3LGPDfxgsMDJ5oA9etpLee3Se0eKSGVQySREFXU8ggjqKkJCqSxAAGST2ryfwzresaNH4Qt7fVf7YttT0N5msPKjAh8qFCjRFQG2k/Kd7NktwRT/A3ibxdr32HUNUjuW0rVLGZ5vtRs0jhnAyq24ifzSAAwIkBYbcnHIoA9Ig1vSrn7L9m1Ozl+2hja+XcK3n7fvbMH5sd8dKvV4h4L6/Cz/rlf8A8jV/SPHuvXOneEDcaorXGp+JbqyuV8qMNLbxtMAuNvAG1BuGD0yeaAPYKK8bm8VeLXvjPD4hMUQ8WtpC2/2OFk8g+p27iw7EEe+6u2+HurajqdjrEGrXkl9Lp2rT2aXEqIruikbdwRVXPOOAKAOuooooAKKKKACiiigAqlqB06OS3uNQjheaFy1sWi3yBtpB8sAFs7c/d5xmrtMmgiuYWhuIklicYZHUMrD3BoA59oo4NehmstMbTpr8tJe3MUNuskwjwF8xjkkYP+8BjpyKuza48dgbqPTbpx5iKisUXzFY4DD5unscHpkCraabBHJE6mQ+UrqoeQvw2M8tk9uBnioho6fYntTd3LISvlFmUmHacqF+XnBH8WT65oAF1YS3E8MNndSPb4EmFUAEgHAJbBOD29PpmzbXkF2rGB8leHRgVZD6Mp5H4imCwj8qVGkmbznDuwcoxOAOq4x0qeGGK3iWKCNYo16Ii4A/CgB9FFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFAFXUZrJLVotRCSRTqUMDJ5nnDHKhMEtxngA8Vz9ppthot5a2mgaFHoVvdF5bkWVrbRbwmAN4GeMHsN3I6c1080EVzC0NxEksTjDI6hlYe4NV002COSJ1Mh8pXVQ8hfhsZ5bJ7cDPFAGS01jbeHHt7bw7/AMS92Ea2YiiSOVJGwSFztwc9Dg88gZzRpdnommXU8eheG4bWWCNYZTaWkUWF+8I8jAI+YnA46+ozojR0+xPam7uWQlfKLMpMO05UL8vOCP4sn1zUwsI/KlRpJm85w7sHKMTgDquMdKAM7StD8MxWt5BpOjabbRXXF5BFZJF5mQeJU2gnIJ+8O9bEMMdvCkMEaxRRqFREUBVA6AAdBRDDFbxLFBGsUa9ERcAfhT6ACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAqNpOnPBcQvp9q0VzJ5s8ZhUrK/HzMMfMeByfQVJLY2k93BdT2sMlxbbvImeMF4twwdpPIyOuKnooAzn8PaLJpL6XJpFg2nyMXe0a2QxMxbcSUxgktznHXmlTQNHj02DT00mxWytnWSC2FsgjidTkMq4wCDyCOhrQooAzrjw9ot1rEWrXWkWE2pQACK9ktkaaMDONrkbh1PQ9zUd14X8P31m1pe6HptzbPMbhoZrON0aU9XKkYLH161q0UAYn9leG9Itbewt9HsIo4pftFtZ21mnyyZ/1ioo+Ugn7/ABjPJFVda0vw7rE4uPEnhyyv5YYowq3lhFPJHvYjaCd3fsP1roLi2gu4vLuYllTIYBhnBHQj0I9e1VzpcBP35ekY+aQsfkbcMk5J5PJJoAyf+JZbi3a28NxiRj/ZzRrFCrRRYz5fXGzHO0HH8qtWltotveWcQ0q3sLmzRoLINbonlxnqkTDgAhc7VOcAZAq3LpUcm4rNLE7Ti4DptyrYxxkEYxxyO9TW9jb20jyxpmWT78rsWcjOcbjzjJOB0HagCxRRRQAUUUUAFFFFABVe8sLPUY0j1C0gukjkWVFnjDhXU5VgD0IPIPUVYooAq6jpdhrFi9lq9jbX9q5BaC6hWVGIORlWBBweao2vhHw3Y2ptrLw/pVtAZlnMUNlGimRfuvgLjcOx6itiigDIv7LQla4iu9NtJ5b8q00Itlke5K4Cllx8wHHzNwOORVTUrLQ725a61rQbW5nW2SIieyjllVJGKmPdzkHuo469c1u3FtBdxeXcxLKmQwDDOCOhHoR69qrnS4Cfvy9Ix80hY/I24ZJyTyeSTQBlTtpreU//AAjqSyyk6dIjRQ70ixnyyc4KY52g4/lVy0j0m31CIDTobC8SL7NCWgVCYgSQiOOCvGdgPHcCrEulRybis0sTtOLgOm3KtjHGQRjHHI71Nb2NvbSPLGmZZPvyuxZyM5xuPOMk4HQdqALFFFFABRRRQAUUUUAFFFFABRRRQBDeWdtqFnLaX9vFdW0ylJYZkDo6nsVPBH1rP0rwr4e0G4efQ9B0zTZnXY8lnZxwsy9cEqASK1qKAKR0XS2gvYW02zMWoMWvIzAu25JGCZBj5yQAOc8CoYPDWhW2nR6fbaLp0NlFMJ47aO0RY0kByHCgYDA8561p0UAVBpOnD7ZjT7Ufb/8Aj7/cr/pHy7f3nHzccc544qGDw7olro8mkWuj2EOmygiSyjtUWF89coBtOe/FaNFAHKa58P8AStT0vS9MsLaz0ywsNQS9NpDZp5Uu3O5NgwBuzycH6GtmDw3odtpP9l22jafDp+/f9kjtUWLdnO7YBjOQDnHatKigCjPo+lTC8a606zkF5GI7syQKfPQDAV8j5gATweKwrfQvCVnYX/8AY/hfTIrSe3YXEsGmxLDOoGQpHy+YpznjK+9dXVOXSrSUzlUMTXCMshiYqHyMFiOhbjqQTQBk6nZeH9Vure31bw9bX62kixRS3NnFIluzAFdu7kZ4HyjjvipY57W6uH1G50BhNbmS0Fz5UcsixZ+YDaS20lRlQDzjg9a0m02F9+Wf5pkmPI+8oGO3T5RUf9j27easzyywySGQwO3yZPUEDG4HJOGyKAJNLi0+DTYoNGit4bKIFI4rZFWOPB5UKOBg547VbpFUKoVQAAMAAdKWgAooooAKKKKACiiigAooooAyj4W8PnW/7ZOhaadU3bvtxs4/PzjGfMxuzjjr0qSbw9otxrMWr3GkWEupwgCO9e2RpkAzgByNw6nv3rRooAr21hZ2UlxJZ2kFu9zJ5s7RRhTK+ANzEfeOABk88VTtfC+gWLXjWWh6bbtfAi7MVpGhuAc5D4HzZyeueprUooAztO8O6LpF1Lc6To9hY3E6hZZba1SN5AOgYqASPrRa+HtFsdRuNQstIsLe9ugRcXMVsiSTZOTuYDLfjWjRQBjR22i21zawadpNvJLYllhW1tkAtN2N2G4CZByVBBI7GqQsfD9tdm7i8O2s189xJc+bBYxLIZIyVMhYn72GPzE5OT0zit6ext7iZJpY/wB7H92VGKOBnONwwcZHI6HvTE02GN9wZ84kHJH8bbj29RxQBkiPTJL0w2/hyGdPl1JZUihAaY5w+CR8/H3uvuOtaWmf2ajXCadBFayvIZriJYRE5cnBdlwCScfe746mgaREpj2T3CKkAt2CPt3qOmSBkEc8qR1q1b20NrF5dtEsaZJIUdSepPqT3PegCWiiigAIyCD39DVef7Pa28k9zMYYYlLvJJMVVFHJJJOAKsUUAVLKay1Gziu9Pu1u7aZd0c8FwXRx6hgcEVP5Cesn/fxv8axPBH/IoWv+/N/6Net+gCPyE9ZP+/jf41V1G803R7F73V7+KwtUIDT3V15SKScDLMQBk8VermPHKWS2Fle3evw+H57O5Etrf3KqYFk2ldsgfClWUsMblPPBBoA34Ps91bxz20xmhlUOkkcxZXU8ggg4IqTyE9ZP+/jf415reT2mp6d4T1TX2TTIY9TlR7ixvZrW2kJSQLKrKy/LI2CCc534yQ3LZfE4k+J1t5N39nii1A2V0kusSFgDG2wPZ7fKjVmA2SE724xnNAHoeoXNnpdm11fTSRQqyqW3O2CzBRwMnqRUkTWs0ssUNx5kkDBZUSckxkgEBhng4IPPY15pFaXTfD3VdT1HVb7UJ7nVTGi3E7GOCJL7CoidOB/EcsemcAAaGuX+oeT4pSDUbu3aPV7GGGSKUhoVcQbgmcgA7jxjBycg5NAHoHkJ6yf9/G/xo8hPWT/v43+NcJdRajpw8Q6fpNzqU8MUtpI4NzLcTxxOf3/lMxLg7QSFXofujOKl8HTBvGepQWEuuHS10+3eGPV5JywcyShmC3B80ZwBlvQY4xQB23kJ6yf9/G/xo8hPWT/v43+NSUUAR+QnrJ/38b/GjyE9ZP8Av43+NSUUAR+QnrJ/38b/ABo8hPWT/v43+NSUUAR+QnrJ/wB/G/xo8hPWT/v43+NSUUAR+QnrJ/38b/GjyE9ZP+/jf41JRQBH5Cesn/fxv8aPIT1k/wC/jf41JRQBnapqOk6HaC61rU4NOtywQTXd35SFj0GWYDNSafdafq1kl5pV9He2smdk9tc+YjY4OGUkGuf8dJeSXXhpdMngt7o6sPLluIDMinyJeqB0J4z/ABCsTxKg8NaXDY3l6Z77WbuSeW5OpSaTA020AIjxBnBOAFiBYtg5JwaAPRPIT1k/7+N/jR5Cesn/AH8b/GvLtQ8QSDwR4a1q51sF4rb97bHUXtGupwoBCuoIkmUhgIHBDknIGK2Na1V9P+IOlyPfvILwRQx6b9teCePJ+Z1t8FLhTkb2ODGFJB7AA7nyE9ZP+/jf41FcG2tLd57qfyIUGXkknKqo9yTgV53Zajq0nxMeC6vrS3uUvnU2kmtTebJaYO3ZY+X5ZH3T5obOQct1UKPt8PgzxVrT6zqUtz9suYbdWuWCW0aTkBUAPX3OSBwCAMUAdw2q6autrpPnzm8KCQqqysiA5wGcDapODgEgnHANaHkJ6yf9/G/xrgJ7Ga08VeMtT0dryTU006FoE+1yuhco/SIsUJ4GBt45x1OWaVqFmLq4Twnrt1q1p/Zk0l/JJqMl39nnAXy/mckxOcvlAVxj7oxQB6F5Cesn/fxv8aPIT1k/7+N/jXm9jp96mkeC4LjX9Ynk1WXzL6c3jB5wbRzs4+4vA+5g5G7O7LUnh1ryC68OXD6rqVy1xqV7ZOtxePIhgj87YpUnDEeWvzsC57saAPSfIT1k/wC/jf40eQnrJ/38b/GvNtB1LV5/iG8N9e2cN2t3Os9m2szSTPbDcIytl5YRFxsPmhueck5xWn4P1Nx4y1XSpdR/tOQB53mjvnl8rL/Kk1u4/wBGYA4VVOHCsxANAHbeQnrJ/wB/G/xo8hPWT/v43+NSUUAR+QnrJ/38b/GjyE9ZP+/jf41JRQBH5Cesn/fxv8aPIT1k/wC/jf41JRQBH5Cesn/fxv8AGjyE9ZP+/jf41JRQBH5Cesn/AH8b/GjyE9ZP+/jf41JRQBH5Cesn/fxv8aPIT1k/7+N/jUlFAEfkJ6yf9/G/xo8hPWT/AL+N/jUlFAEfkJ6yf9/G/wAaPIT1k/7+N/jUlFAEfkJ6yf8Afxv8aPIT1k/7+N/jUlFAEfkJ6yf9/G/xo8hPWT/v43+NSUUAR+QnrJ/38b/GjyE9ZP8Av43+NSUUAR+QnrJ/38b/ABo8hPWT/v43+NSUUAR+QnrJ/wB/G/xo8hPWT/v43+NSUUAR+QnrJ/38b/GjyE9ZP+/jf41JRQBH5Cesn/fxv8ao6dqekau9wmk6pb3zWz+XOLa8Ehib+621jtPHQ1B4vivZ/BerxaWrPePZyCFUbazNtPAODgn6VzXhnU01W9k0uG70fUrFdPeIXOiwmNtOGQBbud75Yg5H3DlD8voAdXp2raLq81xDpOrW19LatsnS2vRI0LZIwwVjtPB4PpUlhe6ZqqSvpeoRXqwyGKVre68wRuOqnaTgj0PNcT4YZ5/FOl2mn65Y6vaaRbSwSNY2nlG3TCqsU7b2/e5UHaAmNhJUcY6Xw9/yMPib/r/j/wDSeKgDc8hPWT/v43+NHkJ6yf8Afxv8akooAj8hPWT/AL+N/jR5Cesn/fxv8akooAj8hPWT/v43+NHkJ6yf9/G/xqSigCPyE9ZP+/jf40eQnrJ/38b/ABqSigCPyE9ZP+/jf40eQnrJ/wB/G/xqSigCPyE9ZP8Av43+NUdO1PSNXe4TSdUt75rZ/LnFteCQxN/dbax2njoag8XxXs/gvV4tLVnvHs5BCqNtZm2ngHBwT9K5rwzqaareyaXDd6PqViunvELnRYTG2nDIAt3O98sQcj7hyh+X0AOr07VtF1ea4h0nVra+ltW2TpbXokaFskYYKx2ng8H0pNP1fRNWurm10rV7W9uLRttxFbXokeE5Iw4ViVOQRz6GuK8Pqb3xBp+n2Wu2eq2emWc9tK+nWnktbRlVRYpm3t+9yAdoCYKElemHeGw7eJtH0/T9cstWttHiljf7HZ+U9tFsCrFcNvb94TtO3EfKEleBgA9D8hPWT/v43+NHkJ6yf9/G/wAakooAj8hPWT/v43+NHkJ6yf8Afxv8akooAj8hPWT/AL+N/jR5Cesn/fxv8akooAj8hPWT/v43+NHkJ6yf9/G/xqSigCPyE9ZP+/jf40eQnrJ/38b/ABqSigCPyE9ZP+/jf40eQnrJ/wB/G/xqSigCPyE9ZP8Av43+NHkJ6yf9/G/xqSigCPyE9ZP+/jf40eQnrJ/38b/GpKKAI/IT1k/7+N/jR5Cesn/fxv8AGpKKAI/IT1k/7+N/jVS5vdMs722s7zUIoLq7JW3glutrzEckIpOWx7VfrA8U/wCs0P8A7C0X/oL0AWdW1rQtAWJtd1m00xZiRGby+EO/HXG5hmm6jr3h/SLWC51bXLKxt7kZglub9Y0l4z8pZgG454rH8cnT7O4stTuPEkmgahbxTJZviIpcFtpMRWRDvJKr8qEOecGq3iG5gbSdE1nXNcfwtq6WzNEMRbGldFLQkSo245HCKQ5wcHg0AdjB9nureOe2mM0Mqh0kjmLK6nkEEHBFSeQnrJ/38b/GquiXV3e6DY3WpW/2a7mgR5ocEeW5AJGDyOe1XqAI/IT1k/7+N/jR5Cesn/fxv8akooAj8hPWT/v43+NHkJ6yf9/G/wAakooAj8hPWT/v43+NHkJ6yf8Afxv8akooAj8hPWT/AL+N/jR5Cesn/fxv8akooAj8hPWT/v43+NHkJ6yf9/G/xqSigCPyE9ZP+/jf40eQnrJ/38b/ABqSigCPyE9ZP+/jf40eQnrJ/wB/G/xqSigCpezWWnWct3qF2tpbQruknnuCiIPUsTgCqv8AbGh/2L/bH9sWv9l43fbvto8jGcZ8zdt68dayfiBcJb6RYvLOtkq38T/b5SohsyuWEku4YKZG3B25LDDKcMMLSxp2q+CNZfV9US3tm1IXEuuafIn2eVwUcTRllKqgIAO7cAVbLty1AHZW+uaDd6PLq1rrVnPpsOfMvI74NCmOuXDbRj61Z0+60/VrJLzSr6O9tZM7J7a58xGxwcMpINclaeLrWw8GXmreI72HUbCzvfLsdRkEaC9GV8uTPEYO8ld42r8u7gVreC5rO9sb3UrTVNO1KW+ujNctpk6ywRPsVQgYHkhVXJOCTzgZwADokQIMDcfqxP8AOnUUUAB4pu8e9Kfun6VTvYp5rOSO1ufssrDAmCBynqQDxnHTORnqD0oAt7x70bx71zXhS5u51vkl1GbVrKKULa6hPHGjzcfOP3aqjKG4DBRnnrjNdDQBJvHvRvHvUdcj8QtY1nR7DS5PD7gTyX6iWIoredEsbu8Yz0LBMAjkGgDst496N4964p/FNxqPizRY9Fnjk0mdD5zAA+c7QtIi7u21QpOP749Ku2vi77VFYBbEi5njne5h83P2bycq4Lbefnwo6ZzntQB1G8e9G8e9cj4e8XX+sT6Z9t0VLC31WyN3bMLwSyDAQsrqFAH3xghmz3C9KzZfHP8Aatv4j0+2ksFnstOmnjl03UxcvERldsgVR5UgPYFu/PFAHdXcQu7SSDzZoC44khfa6HsQf8cg9CCOKz9G0SPSZ7i6mvrzUr65CrLd3nlhyq52qBGiIAMnoo6nOayvDWv6ldXNpYatpq2vnael1BL9r82RwNoYSrtAVssPus4PPNdPQBJvHvRvHvUdFAEm8e9G8e9R0UASbx70bx71HRQBJvHvRvHvUdFAEm8e9G8e9R0UASbx70bx71HRQBJvHvRvHvXIePdVutMtdKFrf31gl1fiGebT7QXM+zy3bCxmOTJyo6KeM1Sj8T6lofhOS/1aW3ZTdMlpceIbpNNaWHAKmTEfyuTuwvlg4AyFOaAO83j3o3j3rjbvx4sGm6FfR2cKQatGJDLeXggjjyARGrlSrSNu+VSVDYPzCrt14q+y+NLfQZLeCMTxh0ee7EUk5OciGMjEu0DLYYFcjg5FAHS7x70bx71xEPxHsJ/GQ0RJNOKtdNZgDUVN35qgkk223Ij4I3bs9DtxzSSeM7+50XXr1dKks7XTJZLeO5F0hed0k2kopRgFx3YdcgA43UAdxvHvRvHvXn0uq6ja+Otdv9UBXStHso5I/J1GTJDKxIMGwRsWI6scrgYPJxsW/ibUYZmt9e0mCyne0ku7Zbe9M6yKmNysTGm1huXoGHPXigDqd496N4964e08b6vdaXot0fDPky63LstYJL4BkTyTIHkOzCjKkYXccYOM/LUuj+Nr3UbrTFu9FS0gvp57TzBeeYyzxb92F2DMZ8tsMSrdMoKAOz3j3o3j3riNN+JGn6l4tGjwS6cyvcS2sYj1FXuhJHu3F7cLlE+VgG3E9MgZrS8O+Kv7e1LUbN4Le3kspCphF2HnjAYgGaIqDGWxuXG4Fec0AdLvHvRvHvUdFAEm8e9G8e9R0UASbx70bx71HRQBJvHvRvHvUdFAEm8e9G8e9R0UASbx70bx71HRQBJvHvRvHvUdFAEm8e9G8e9R0UASbx70bx71HRQBJvHvRvHvUdFAEm8e9G8e9R0UASbx70bx71HRQBJvHvRvHvUdFAEm8e9G8e9R0UASbx70bx71DKjSQuiSNEzKQJEA3IfUZBGfqCK4uxn1PT7nWNSbxFqOr6XplvIhju4rUCWdRubaYoUOFxt6nJJH8NAHdbx70bx71xlnqOo6PeabPresNeW2p28ksyyRRoloyxiT92UUHZjcPnLHp83XNXw548svEXjQW1lr2m3FtcWTSwWVvcxSOhDjBbaS28qSSvQDHGQSQDvd496N496jooAk3j3o3j3qOigCTePejePeo6KAJN496N496jooAk3j3o3j3qOigCTePejePeoZUaSF0SRomZSBIgG5D6jIIz9QRXLeHIdUfxDeSHxLqGqaXaqbcrdxWoEk+RuKmKFDhPunk5JI/hoA6/ePejePeud8Nz3slxrdre38159lvzFDLKkasqGNH2/Iqg4LHGRn1zVDSrfVT4vljj8T6lqNhYqVu47qK1CGVgCsamOFGyoO4891HrgA7HePejePeo6KAJN496N496jooAk3j3o3j3qOigCTePejePeo6KAJN496N496jooAk3j3o3j3qOigCTePejePeo6KAJN496N496jooAk3j3o3j3qOigCTePejePeo6wvEU15a6hostpfzQRyX6wT26pGUmVlb7xZSwIxxtI980AdDvHvRvHvXO+K57yztrC5sb6a2239vHJGiRssyPIqFW3KSOCeVKnPeqXi681K31CyihutV0/TpEcyXek2AvJfN42oyGKXahG47tnUAbh0IB1+8e9G8e9Y3hjVJtY8OWt7dJsmfcrjbtyVYqTjJxnbnGTjPU1q0ASbx70bx71HRQBJvHvRvHvUdFAEm8e9G8e9R0UASbx70bx71HRQBJvHvRvHvUdFAEm8e9G8e9R0UASbx70bx71HRQBJvHvRvHvXO+MtRutN0NJLGaSB5LiOJ5IFR5lRj8xiR8h3x0XDE84VjgHP8PXl3r3hS7ZvEF1EIrl0i1FY7dLlY1IP71GjKI/UFSikDGVU0AdlvHvRvHvXna6vq+k+Fmv59burtNTvY4NPnvoId8ETceZtijTczAMyrgnlB6iui8J6gl7ZTxrq95qbwSYY6jZfZbmPIyA8flx8dwdi5Hr1oA6MHPSlpqfd/GnUAIfun6Vl65pS65otzpsl1cWiXC7GmtiocDPIG5WHPQ5B4JrVqK6uPstrJN5UsxQZEcK7nc9gB/jgepAoAzdG0ufSrZobjV7zUxkbDdRwJ5YAxtUQxoMfUGtGqGia8mtfao2sbvT7qzkEc9rd7C6EqGU5jZkIIOeGPvitWgCGqWoaTBqVxYTTtIrWNx9ojCEAM2xkwcjphz6dq06ydf19dBitCNPu9QmvLgW8NvaGMOzbWbrI6KBhT1NAFaz8J6Tpv2JdMt/skVncS3McUR+UyShg5Ocn+MnAxjjtxVPRvDYt9d8Q6jcQtAdSlEcaiXdiMIAWA6KWYscfTPNWIvGMNxFpbw2U6Ne6g1hLDcMFe2kVXZgdu5WwUx8rEHOQa1rfWtKu9UuNNtNTs57+1GZ7WK4VpYf8AeQHK9e4oAzbbwvY20Gmwq87Jp1m1nEGcfMjKqksQBzhByMd6zrTwDaWsHk/2rqUsX2F9PCOYQqwMOEAWMAbex6n+ItxWrqPizS7XTtZmsLy01C70eB5bmzhuVMkZVSQrgZKZweoo1fxPbaXoF5fL5U91aWYvGshOA+0jjPBIBIIzjtQBLBotvb39rdo8pktbQ2iAkYKEqcnjr8g/XitCqEHiXR7y1vZtP1KzvWsAftUdtdRsYWAOVb5gEPB+8R05xU51vTUkginv7WCe4CGOGSdA7b87QBnnODjGc4OM0AWKKxbzxjpcOuWmk2V3aXt5LeC1uIIrpTJa5jdwXQZI+5jBx1roKAIaKmooAhoqaigCGipqKAIaKmooAhoqaigDM1DSoNSuLCad5FawuPtEQQgAtsZcHI6Yc9Mdqq6z4ei1i8srv7bd2VxZlwklsUyyOAHQ71YAHA5XDDswyan1/X10GK0I0+71Ca8uBbw29oYw7NtZusjooGFPU1Bb+MNJOkz3+rzLoaWs/wBmuU1OaKIwyYBClg5QkggjDHrQBSl8Fwv4bg0OPWNThtIozA2x4i00J48pt0ZGMcbgA+B97k5tah4Zh1C8s5JL+9jtbWSOX7CjoYZXjIKFtylxggHCsoOOc81p3Os6ZZratd6jaQLeuEtTLOqidiMgJk/MSOwp8mqafFqcWmy31sl9MhkitWmUSuo6sEzkgeoFAGZB4eW11iS9tdSvobeWQyyaejR+Q0h6v9zeM9SA4UnJI5OWN4Wsm0C+0gy3H2e+mlmkbcu8NI5c4OMYyeODxWmNZ0s6wdIGpWh1IR+abLz184J/e2Z3Y98VBceI9KgjuDHfWtxLboXkgjuYg6gNtJO5gBhsjkjkY60AQv4esZrnUpbgSSrqcKQXEbN8u1QQMYwRwx7/AJVVs/CccDyyXuq6hqcrW7W0Ul20eYIm6quxFBJwPmbcxwMmmw+NLS78dz+G7N7GWS0iD3Ob5ROpIzhYcHcAMbiWBG4cHNauneIdF1eK4k0nV7C+S2OJ2trlJBEfRipO3p3oAqJ4ctEj0ZRJNjRv+PfLD5v3Zj+bjngnpjmorXwpY2n9n+XLcH7BdzXcW5l+Z5d+4Nx0/eHGMduTV4eJdCMTyjWtOMcbFXf7WmFITeQTng7Pm+nPSnWviHRr65gt7LV7C5nuIftEMUNyjtLF/fUA5K+44oAp2nh5bHVZLm11K+S1d2l/s4NH5AkY5Zh8m/kknbv25JOKSz8Ox2uvSarLqF7dy+UYYIrhkK2yMQWVSFDNkqvLsx44I5rSj1jTJdWk0qLUbR9RiQPJZrOpmRf7xTOQORzinWmqaff3FzBY31tczWj+XcRwzK7Qt/dcA5U+xoAkoqaigCGipqKAIaKmooAhoqaigCGipqKAIaKmooAhoqaigCGipqKAIaKmooAhoqaigCGipqKAIaKmooAhoqaigCGipqKAIHXcjKGK5GNw6j3qlpuj2umaJFpUIaS3jj8smU5aTP3ixGMkkkk+pNX7q6hsrOa6u5FighQySSN0VQMkn8KztH15tWV2k0jUtPTb5kL3caYmT+8NjMV6j5X2t/s8HABT0rwtBpd9Hctf3t79niMNnHdMhW0jOMqm1QTwqjLlmwOvJzf/ALKg/t7+1t0nn/Zvs23I27d27OMZzn3qvp3ieO91Kaxu9NvtMmSE3Ef2wR4miBwXXY7beccPtbnp1puleK4NVvooP7PvrOO5jMtlc3KoI7xBySm1yy8EHDhTg5A4OADXoqaigCGipqKAIaKmooAhoqaigCGipqKAIHXcjKGK5GNw6j3qppOmQaNpNvp9oXaKBNoeQ5Zz1LMe7E5JPqau3V1DZWc11dyLFBChkkkboqgZJP4Vn6Nrj6xuL6RqOnoV3wveRoBMnZhsdtvUfK+1vbg4AJLLTIbC5vpoWkZr2fz5AxGA2xUwOOmFHr3puk6VBo9iba3aSTdK8rySkF5HdizEkAdz+WKNI1xNWW+zZXVnJY3BgliuAhYkKGyPLZgQQwI5z7A1BpviQX2qfYLrStQ0yZ42lt/tix4uEUgFl2OxXG5flcK3PTg4ANSipqKAIaKmooAhoqaigCGipqKAIaKmooAhoqaigCGipqKAIaKmooAhoqaigCGql/psOotamdnX7LcLcJsIGWUEAHI6c1o1m6nrS6Xfafby2VzKl9P5AuIvL2QuQSN4LBucH7ob3xQAup6ZDqttHDcNIqxzxzgoQDuRw4HIPGQM1Fq2lS6nGiwarfaay5Bks2TLqeqkOjDt1ADDsRk1HrfiQ6PqFnYW+kX+q3V4kkiRWZhXaqbdxJlkQfxDoSak1XXJtMs4Z00TUr53UvJDbeTugUDJLF5FXjphWJPYEAmgCxYWNvpmnwWVjH5VvAgSNMk4A9zyT7mrFNsL6DU9Ot76zcvb3MSyxMQRlWGQcHpwasUAQ0VNRQBDRU1FAENFTUUAQ0VNRQBDRU1FAENFTUUAQ0VNRQBka1osWtQwhri4tLi2l863urYr5kL4IyAysp4JGGUjnpVJPCVr/Yt9p9ze3ly2ovuvLuRkEs/AXB2oFA2qFwqjj3Oa1dY1aPRrJZ3gmuZJJFiht7cKZJpG6Ku4hR35YgDHJqraeIxe6Ld30Ol332izdo5tPbyhOrqASud/lk4IOQ+MHrQBNqWlQalp4tHZ4AjK8MsJAeF1OVZcgjII7gjsQRxUOj6ImkmeWS8udQvLkgzXd1sEkgUYUYjVVAA9FHvk805vEtknhi31x1mW3uY43ii2Ayu0mNiAA4LEkDrjPfHNWdK1GXUrZnuNNvNMlRtrW94E3D0IaNmRgfZjjocHigC4n3fxp1FFABVbUVkfTZ1ihedmQjy45jC7DuFcEbWxnByOccjqLJ4Bpm8+1AHN+C9PvLFNRa4sbrT7WedXt7a/uEnuRhAGaSVWfdkjjc7kAdQMAdPTN59qN59qAH1yfj7SrrVLXSTa2F9fpa34mnh0+7FtPs8t1ykhkjwcsOjjjNdTvPtRvPtQBw+naFrAsdCEtk8KWmsyXXkyyRebBAySgeYUO1n3PyQWJzkljk1V0rSvFTeMLS+1W2vBAiXds4DWawWxkKsskKp+8KHbgl2L7iPkwSR6FvPtUc91HbW8k9zJHDDGpd5JGCqijkkk8AUAcRb6RqzfD+88OyeHzFd2uly2kN4ZoSl07LjMZDbgGIBbeE5x16iGbwxrUPgrX/D5s/7Smv7Rmiv5ZY1a4dlC+VMOMEYwCq7NoHAxg98bhBCZS6CMLuLk/LjrnPpRFcpcQpNA6SxSKGR0OVYHoQR1FAHGXWlarrBv5U0WTSxDo02nQwyyxE3LtjaV2OQEG3jdtPzdBVzS/DcsfjKPVryyjDQaLBZwXDFWaOTc/mKOcjjbz0PrXVbz7Ubz7UAee2Oi63v8L6fcaAVGh3xkuNReeHZKDFKpkjAYudxYbgwU/N/Fya9Fpm8+1G8+1AD6KZvPtRvPtQA+imbz7Ubz7UAPopm8+1G8+1AD6KZvPtRvPtQA+imbz7Ubz7UAc94v0GXX5NFhVZzBBqAmuHt7prd0QRSDIdGVhyVHynPPpmqGu6HqGm2mn2PhK2uYbUySfaZ7NoHugzDh2e63BlJzvb55DxgHmuw3n2o3n2oA86n0HWovBWiR2uk3i6zY2jWQaGa1ZRkAESrIdjQNtBO394BjCgk41dW0rVR4usLzS7KYNK0IvZUkgazdEOf3iSfvVdctsMXcjccDA7DefajefagDgrPwxq0HjR2upNWmsxqL6hDJG9mtp8wIw3yfaNwBK4BKkAfMBwHr4SuofBPiW2t9OjTUtUvric7WQNODLlCWzj7mOCeK7refajefagDlb/w9qF9deKUjxAup2MMFtMXGCwRwc4yQPmHbvxmq8dhqmq332p9Ck0ZbXS5rFY5ZYWNwz7doTy2YCNdpwWKn5vujmuofVrKPUo9OkvbZb6VDJHatKoldB1YJnJA9atbz7UAcTZeDvsem+CLOHSoI4dIk825jATEL/Z3Xd15be3UZOTn3pmj+Fr6wbQz9gSEwaxe3d0UZB8snnBXOD8xO5PU+vSu53n2o3n2oA4LRfC+rWPi0fbpdXuLaC+uL2GbfZrafvN3BwguC+Hxg5XjO7oK0PDWlanpviq8QWdzbaQsblFupIJFWR3DH7OyfvfLJ3FhLjB27QAK63efajefagB9FM3n2o3n2oAfRTN59qN59qAH0UzefajefagB9FM3n2o3n2oAfRTN59qN59qAH0UzefajefagB9FM3n2o3n2oAfRTN59qN59qAH0UzefajefagB9FM3n2o3n2oAfRTN59qN59qAH0UzefajefagB9FM3n2o3n2oAfRTN59qN59qAM7xLpUmueF9S0uGUQyXdu8SSEA7WI4PIPf2NYXhnT9Wt9alufsuq2Fp5DLNb6rqQuvPnyCrxYkkEaAbhgbOo+Tjjrt59qN59qAOK0rQb2fV9V3adqGmWd3ZvbztqF6t3JNIxOGiffI6xgFsKSo+bhAc1PpVlrF3qGiQ6lpb2EWhxnfcGaNku5PL8seUEYtswWJ3qp+7gdcddvPtRvPtQA+imbz7Ubz7UAPopm8+1G8+1AD6KZvPtRvPtQA+imbz7Ubz7UAPopm8+1G8+1AGd4l0qTXPC+paXDKIZLu3eJJCAdrEcHkHv7GsPwvp2qQa7JcPbatp9l5BSeDVNRF35824ESRASyCNQN3A2ZyPlGOOt3n2o3n2oAyNIs7uwvNenlgJFzeedAodf3iiGNfXjlSOcVl+FG1W71SW/8AEegalZahJEVEtxNbNBAmQfJiEczt6EsVBYjkgBVHV7z7Ubz7UAPopm8+1G8+1AD6KZvPtRvPtQA+imbz7Ubz7UAPopm8+1G8+1AD6KZvPtRvPtQA+imbz7Ubz7UAPopm8+1G8+1AD6KZvPtRvPtQA+imbz7Ubz7UAPrI1+xuL2TSjbR7xb6hHNL8wG1ArAnnr1HStTefajefagDm/Gulw6paRRzeFI9ffDLDJmEPaSEcOGkKlB/tISwwMA1W1zS79vCunWN7YalrlxHAsc0mnagLdmlCYLSB5EWSMnOQ27PGUPNdbvPtRvPtQBV0aG7ttDsYNSaN7yOBFnaJQFLhQDgAAYz6AfQVdpm8+1G8+1AD6KZvPtRvPtQA+imbz7Ubz7UAPopm8+1G8+1AD6KZvPtRvPtQA+imbz7Ubz7UAPopm8+1G8+1AD6KZvPtRvPtQBgeNNOu9T0m3ht7I39ulykl1axsizSouSPKZyFVw20htykY4YHFQ+ENLvtP0C+t7m2mtopZpHtbe6eN7hEZRnzZEJDsW3HcWZiCNzE5rpd59qN59qAOQufDt/N8PdEsvLlW900W0skEMqq7mNcOivnAYgkA5HOOR1F3wfY6haJetdRahaWUrqbWz1O9+13EOBhy0nmScE8gb2x7ZwOi3n2o3n2oAfRSKcjmloAQ/dP0rK1661Cz0aabRrCTULzhY4Y2jB5OC3zsqnAycFhnGM1rHoaj2H0oA4b4e3Er6p4khl0rULErexs73zwMzuYY92TFIw3cZ9MMMdwO3pkVnFBJLJDBHG8zb5WRQDI2AMse5wAMn0qXYfSgBtcv43eEppFtqcvlaTdX6xXpLlEdSrbEcj+FpNikE4bO05Bweq2H0qO4tIru2kt7uCOeCVSkkUqhldT1BB4IoA5C8Oj+DbPWY/C+j22n3VvYC7ZLaBYoG5YBiikAkbSScZwAM+mXr9xqFho+saXc61NrcV1oNxdmWeOFWgIAAx5SqNjbjjIJ+U/Ma7jSvD+k6FBJBomk2WmxStukjs7dIlc4xkhQMnFMsvDWi6da3Ntp+jafaQXZJuIoLVEWbIwd4Aw3HrQByep3t3drqNomstpVtpekRzlFjiKz70fJkLqTsG0D5Spzn5vSPRby/vU03TINYbR4LHRbW8BSOJvtOQQ2/wAxT+7XaM7drfN94cV2V54e0nUbi2n1DSbG6mtARbyT26O0IPXYSMr0HSi68PaTfLaLe6TY3C2LBrUS26OLcjGCmR8pGB0x0oA5l9T14eIfElzFe+dYaRArW+nxQLunkMO7DPjO3OMAYOT1xxVXw7q/iW50u/kuZJ4xNYC5s7rV3swglIP3RbO37nlTl8kerV3cdpFFNLLFBGkkxBldVAZyBgZPfA45qrY6Bpel/af7M0uys/tbF7j7PbpH5zHqXwBuPJ5NAHD3mo6pd+DJbeTWdW07UrHUrWC4ldbZZ13vHwWjVomUh8jaBxgMOoPoUEbRW8cckzzsqgGWQKGc+p2gDJ9gB7VTg8N6Na6PJpNto9hDpsoIkso7ZFhfPXKAbTnvxV2C1jtreOC2hSGGNQiRxqFVFHAAA4AoAdRTth9KNh9KAG0U7YfSjYfSgBtFO2H0o2H0oAbRTth9KNh9KAG0U7YfSjYfSgDifiRZ/wBoR6Ba/wBm2OqeZqqj7HqDbYJf3Mpwx2P9funkD61Qlt7vwnoC6ZaPPZ3t9NLdRWPh+K322kaqNyxtchYxGpwSSoJLcAV6BLZxTvE88EcrQvviLqCUbBG4Z6HBIyPU1BqGjafqwhGq6da3ot5BNCLmFZPLcdGXcDhh6jmgDibrxFrT+E9B1tLuRYfsP2vUFs0gMrDap3lJDgxDncEIfkbT2rR1LWdRsvFtpM11KdFm8mFfsywtGkshx+/VsSgtlQhQkDJ3DAzW9N4Y0S4tYbW40XT5be3m8+GJ7WNkjkyTvUEYDZJORzzU8+i6ddalb6jdadaTX1qCLe6khVpYQeu1yMrnPagDi7HXvEV54zlK29+unw6i9lLG5s1tUjA+Vvv/AGjzCdpAxtIbhejVXmv9buPCHifU9R1KO4hiuZ7e0tDaRFERJtuXyp3txj0wBkE813Z0LTG1hdWbTLM6kqeWL0wJ5wX+7vxux7ZqQ6XZm1ltTZW5t5mZ5IvKXY7MckkdCSeT70AcHcfa9K8VeL9Zivbi6NrYQTRWUkcRi3bHKnhA/wAuD/F3Oc8YvxX+qaTe/ZX12XWVutLmvRLNFCpgZNu0r5aqNjbjgMGPy/eNdilnDHPJPHBGssoAkkVQGcDoCe+MnFVLDw7pGlJcJpekWNkt0d0621skYmPq20Dd1PWgDirK78UNpPhQXfiLN3rUu66ljs4gsaG2Z9kYI4OVB3Nu5zxt+Wl0DWNea60KXUNZe7jvL67sZITbxIrJF5u2QlVz5n7sZIKpyfkFd4NOtQLcC0hAtf8Aj3AjX9zxt+X+7wSOO3FIml2cfleXZW6eTI0kW2JR5btncy+hO45I65PrQBxGh6/4j1DxUZJbW/TT/t09pKkzWa20SpuClMP55kO1SQwwQxwBwa0fDWrak3iK70/W7iWWSZZLi1IWFreSIOADC0fzgAMoYSjO4/KSBXRDQtMXWG1ZdMsxqTJ5ZvRAnnFf7u/G7HtmltdF06xv7m+stOtLe7vCDc3EUKpJOR03sBluvegC1RTth9KNh9KAG0U7YfSjYfSgBtFO2H0o2H0oAbRTth9KNh9KAG0U7YfSjYfSgBtFO2H0o2H0oAbRTth9KNh9KAG0U7YfSjYfSgBtFO2H0o2H0oAbRTth9KNh9KAG0U7YfSjYfSgBtFO2H0o2H0oAbRTth9KNh9KAG0U7YfSjYfSgCKaKKeB4rhEkidSro4BVlPUEHqK4nwvoXh61mvvFelaLY6Xb+TJDaNY2aRM0CnLS/IPmLlcj/ZC+prumi3qVdQysMEHkEUyC1jtreOC2hSGGNQiRxqFVFHAAA4AoA8v8I20eka5pl7PYRWi32myyLe2aqZNQGFffdAMT5gXkf6wZY/OCQpd4NtVsvEei3rWMMC6pazNFqFqq+bqOQHDXWGyGCgkf6wZP3kyFPoOn+G9G0i8nu9K0ewsbm5OZ5ra2SN5ec/MygE8880WHhvRtKvp73TNHsLO7uSTPPb2yRyS5OTuYAFueee9AF+inbD6UbD6UANop2w+lGw+lADaKdsPpRsPpQA2inbD6UbD6UANop2w+lGw+lAGfrl1HY+H9Qup5ZYY4baR2khGXQBTyo9R2rzvQLO60C4uUh0y0sL+40N5bc6co8u+dcHzpsNkS5YcHcPmOJHzgeqFCRgjIrO0vw5pGiNM2i6RY6e05zMbS2SIyH1baBnqetAHJaNZ6Xpur+HptCjhjfUrKSW/mj4a5QIp82Uj7zbyPmY5+Y88ml8OWFvpnj1vL02x0w3ljJLENMm81LxRIv76diqMZPmGMhvvN85ziuwstD03TLm5uNN0yztJ7pt9xLBAkbTN6uQMseTyaSw0HS9KuLmfS9Ls7Ka6ffcSW1ukbTNknLFQNxyTyfWgC5RTth9KNh9KAG0U7YfSjYfSgBtFO2H0o2H0oAbRTth9KNh9KAG0U7YfSjYfSgBtFO2H0o2H0oAbRTth9KNh9KAG0U7YfSjYfSgBtFO2H0o2H0oAbXn/xOvdTGj6lB/YWoXOlxWTSG4tpbcI8hB4cPKr7V4OApySPTB9C2H0qO4tIru3e3uoY54ZF2vHIoZWHoQeCKAOPvLSy1/xNa2vibT43sxpX2iGzvlR1WTeA7EAld6jaARnG44PNZWk21t4jbRrLxRBFqGntpcs1vFeqJEmIlAEjBurCPYQSM/OTnmu81LRNN1m3S31jTbS/hjYOkd1Asqqw6EBgQD70aloem6zZraaxplnf2ysGEN1AkqAjodrAjNAGb4LnluPCNnJK7SL84hkdixeIOwjYk8nKBTk9a3aVYtihUUKqjAA4AFLsPpQA2inbD6UbD6UANop2w+lGw+lADaKdsPpRsPpQA2inbD6UbD6UANop2w+lGw+lADaKdsPpRsPpQA2inbD6UbD6UAcr49UXOk2WnhY5pL2+jjS2uE3QXOMsY5eRhCqn15x8rfdPMRLFb+B9R0drcWE7aulr/Z8Kg2sTuyMIFIYAwsD833SQzfICdlek32m2mqWUlnqdnBeWsoxJBcRrIj9+VOQagj0DSodGbSItKsk01lKGyW3QQlT1GzG3B+lAHF+H9Pgh0nxFos2jafbP9qT/AIk/neXp+XRdqI6g5VyCWBjUkscx4OW1vAMEdnZapZR28dm1vfsr2dsQba2JRDshxj5MEHlVO4t8q9K3Lfw5o9po8mk2mkWMGmyhhJZx2yLC4b72UA2nPfjmrFhpdnpVklnpdlb2VrHnZBbRLGi554VcAUAW0+7+NOpEBA5paACsvxLfX2m+F9RvdIt/tN9BbvJBD5LS73A4GxSGb6Dk1qVFdRzS2siWs/2eYj5JdgcKfcHqPbj6jrQBh+D9Yn1nT55bvU7W+mim8t0h02awe3OAdkkUzs4bkEZ25BBAwc10NY+g6Lc6bNe3mqXsd7qF86tNJBbmCIBF2qqIWYgAdcsSST0GANigArO1xtQj01pdM1Cx09osvLNfWrTxhAMnhZI8fXP4Vo1z3jLw7e+JtJgsbLUYLKNblJp0uLVp0uEU5EbBZEO0sATzyBg8E0AZuheKtWHhmLUtft1vZ7+5KaZbadafZ5biPGVJSWZgpIVm5cALjPPFO1XxfMreHrnTLe8kS9vJreexjiQzMyRSExEk7VIdMbtwXj72Kfq/g678S6LFbeJbjRr+6tbkT2rHRybdflK7ZIXmfeMFujKemMYzTm8GTxabo0el3tjp1xpMzzxi201UtmZo3Tb5IcFU+fOA+7j73egB1z4/sLPTIrm507UlnkvfsDWKQpJPHPtLBCEYqcjGCGI+YEkDJF/UfEv9mTWi3Wkaj5Fw0cctyoiMds7sFVX/AHm48kDKBwO5qnB4QkWSyubvUBNexakdRuZUgKJM/lNHtVSx2KFK45Y/LznOaz9f+Hja34jk1b7bZCTdDLbvc6cJ5rZ42BCxyFxtjOPmVQGJJ+ftQBXv/FWr23ijxG7x31rp2iWqsp8m2lgkJTduYeYJSx/hAKrgHdg4ra1XxtbaXfXloNL1K9ksbaO7uWtok2RxNu+bc7qDjYcqMt6A84XUvCR1GDxCjXwjOtQRxEiHPk7UK5xu+bOc44qW68L/AGm41qX7Zt/tWwSyx5WfK2iQbuvOfM6cdOvNAEl34ohg1JLKz0++1GTyknnNoiEW0bZ2s25lJzhvlQM3HTpWNa+NJrXWtZt9Qs766tbbVUtVuYoUEdqrxxbQxJVnyzn7ocjPOBitJ/Dl9b6wt/o+qx2hlt47e8Sa180SrHnayfOux/mIyd46fLxRL4T8y31WL7bj+0dQivc+V/q9nlfL15z5XXjr04oA6KiiigAooooAKKKKACiiigAooooA5jxz4in8OWWnSQahpumLd3ot5b3U0LQwKUdskCSPuoHLDrUOjeK7+48Oy30mnya2yXDRwTaTEsUd5GAD5yCaXaF5I/1h3Yyuc1tato/9qXmlz+f5X9n3f2nbs3eZ+7dNvUY+/nPPSs7xd4SHik2IlltGhtnYyWt/ZC6hlDDG7YWUCQfwsdwGT8poAfJ4ys/L0lrSzvbw6rCbiJIUXdHEACzsGYZxuHyruY9gatN4ihHiQ6NFZ3c0kcSSTTxqpjhDkhAwLb+dp5CkDHJFYV54DubvwXYeHm1GxkjtEEXm3GmCTCjhHjHmDy5lGMOCQDk7emL2seEn1bXNKvWu7cJp0iyK8lmHulIxlY59w2K2AHBVtwyMjNAFlPFds+uCwFleeQ05tV1Aqn2dpwCTEPm354Izt25GM54rPk8d28+n61PZWd9Gmku8M11NbAxGRX2lVHmKX9eCAAeSDxTLX4fWVl4xk1q3g0jbJO1yXk0lGvEkb72253ZCnngqSMkBgMAXH8I7/Cup6L9tx9vuJp/O8r/V+ZKXxtzzjOM5GaAM3/hJNQHjzVo71b6z0bSLRJXfZbtBIGUsWb5jNnj5duBwcjpWtZeLY7hpUvtJ1HS5Vt2uoorxYszxL95l2OwBGRlWKsMjinXPhaC9uNbN3Ozw6vbR27xquDGFVhkHnJO7PTjHeq8HhjUZ7h7jXdWgvZktJLS1a3svIEaSY3M4Ltvc7V5BUcfdGaAK9r8RLK803Tb630fWDHqspisVNuoec+UZN2C/yr8pXc+0ZGfu4ap9K8dWerXVjCmm6lbLevLAk1xGios8e7fEcOSSNjfMoKHHDGrMfhnZF4cT7Xn+xO/lf679y0Xr8v3s9/T3qCy8IfY/7K/07f8A2dfXN5/qceZ53mfL97jHmdec46DNAFi28V2tzrK2S2d2kEkz28N+4TyJpUzujX5t+RtbkqFO04JqbRPEUOuvcG1tLqKGKR40uJVXZMUbaxXaxIwwIw4VvQY5rG0r4fWOj+K5dVs7fSFjeaScP/ZKfbA7klh9p3ZKZJ427ucbsVd0rwq9h4vvddlubVmuYvKxbWQgklGQQ07hiJWAGFO1cAng5oA6OiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooApazqcei6JealNG0iWsLSlE6vgZwPr0rC8O+IrrUIb95dU0PWJbaMu9ro7kyWzj/li/wA77jwRu+Tkfd9N7VtMtta0e70y/TfbXcLQyr6qwwaytK8OXtvqyajq+o291Pb2zWlr9jszbBIiVJ3Au+5vlHI2gc/LzQBj+F/G8uq6nbWt1qmkahJeRu3kadG6yWEigMYZsu2WxkciM5U/L1xY8FeItW1q6mj1y6tYblELNpv9kXFnNEN2A2+WQiVeMblXGe/atSy0PUP7aXUdb1SO9e3jeKzSC18gRqxGWfLtvfCgbhtHXCjNJp+g3ya8NV1rU4r6WCF7e1ENr5GyNmBJf523udq8javBwozQBvUUUUAFFFFABRRRQAUUUUAFFFFADJhI0LiB1SUqdjOpZQexIBGR7ZH1rk7LV/EFnqepvrOoaXfaZplqz3ElnpsluwlxuCAtPIDheTwMbl98dc4YowQhWI4JGcH6VlaX4fgsPDX9kXD/AGoSI4upWXaZ3fJkcjnG4knHbNAGdpms60mpafHrn2ExatE0ltHbROj2zhQ/luzMwk+Un5gE5X7vPEXhzxFf6n4iurJ77SdWtYo2aSfS1YLZyhseRIxdg74JOQEI28qMirOl+Gr61v7abVNWW9i0+FoLBI7bymQEBS0hLsJH2jGQEHJ+Xnh+n+H79PEn9r6xqVveSQ27W1sLey8giNmDHzGLtvb5R0CL1+XngA6CiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACuW8Za/faGsR07UNJhndS0NjeI0k+oOP+WUSq6kHp82H6/d456msLxPoeoa9YTWFrqVtbWV3EYLuK5sRcbkOQdnzqFYg9WDjgfL1yAR6jqWsXGrRaVoZs7S6W1F3cS3sLzqgLbVjCq6ckhvm3cbehzxi6t4z1UaTo19p4isbe6WX7dcyaXcaglsycYIhZCq7g3zsQMDnFbmoeH7z7Vb3mgajFY3cVt9kd7q2NwskWQRkB0O4EcHOOTkHsj+H9Qs9CtNM8P6tHaLChjle7tftBlB+83Dph85OeV5OVPGADZspDNYwStPDcF41YzQLhJMj7yjJwD1HJ+pqeqmlabDo+kWmnWhYw2sKwoXOWIUYGferdABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQBi+K9XudE0M3dp5aMZUje4miaWO2VjgyuqkEqvfkAdSQATVTQvEs1x4ckv9SMN1icw209jGyJqA42PEjMThs4HzMOC27bzV7xDokus29qbW6S2urK4W5gaaHzoi4BA3x7lLD5iRhgQQDniqGn+CNOGm3dt4itdO1c3l2buaJrBVthJgDKQsX29MkkkkknPNAEEXifVP7ElaeC1/tJ9UbToFQN5andgM3OW2rknG3dt/hzxpaDqV/JfX2la3JazX1lskM1pE0Ucscmdp2MzFSCrAjceme+BmWHw803SdHurLSEtbCSTUPt8EttZpGInDbkUqPvKB8vUZUkDFa+h6PdafLd3mq3sd9qN4y+bLDAYY1VRhERC7kAZJ5Y5JJ9gAa9FFFAAfumql3eQWFnLdXs6wW8KF5JZGwqKOpJq0fun6VznjSwutR8MSxWALSxzQzFAu4uqSK7ALkbjgHA7nigC/pGvWWtwvJYNcDZjdHc2stvIAeh2SKrYODg4wcHHStDJ9TXK+ELnU7i8vfPvdT1DTAqG3udWsBaXHmfNvTYIoiUACkEp1J5PbqaAFyfU1m614gs/D9stxqIvTEckta2M9ztAGSW8pG2j3OBWjVDXo3m8OalFCjSSPaSqqKMliUOAB3NAEGmeKLDV7P7VYpqJhLIqtNplzDu39CA8YJXnlhwOpIrTjuYpnlSGZJGhbZIquCUbAOD6HBBx7iuT1SyupPBHh+CO2maaG409pI1jJZAroWJHUYwc+mKXwxpVhpnjLxG0ejLaXlzcCZbtLEos0RjTgTBdp+cMSu7Oecc5oA6/J9TRk+ppKKAFyfU0ZPqaSigBcn1NGT6mkooAXJ9TRk+ppKKAFyfU0ZPqaSigBcn1NGT6mkooAXJ9TRk+ppKKAFyfU0ZPqaSigDP1nX7HQLeGbUnuAJ5RDElvbS3Du5BOAkasx4BPTtUmlaxaa1Zm50+SRkVzG6ywvDIjDqrI4DKfYgdRWH400281ObQI7Ca7tWTUg73VpGjvAvkyDd86soGSBlgRz64rF8VaGLDSrXS4tPGrJf3Dy3d5qWmyakHm2gKXhiKAE4wHO1EC9uKAPQ8n1NGT6mvK7+znfwT4aup9PvbjU9Pt/KS1udLmuFMwAGCFIeF8rhZ87VBJyQRnX1q0mh+IGl6lb2E1xdzCKF45LGSURRg5Zo7pDthxk7lfIk2gAc5IB3uT6mo7i4FtbvNJ5jKgyRHG0jfgqgk/gK83stG1FfiW9ze7EulvXlS4j0OZpJbYghYze+Z5YQAjMZUHK8KThiDwuYfBvim6TTrhtUvry5C70ZpDH55KqgPRD97C8EnPNAHcDXkfxE+kQ2t1K8UavNcIF8qHdnarZbdk4PIUgdyK1Mn1NcDc+GG/4SHxbqGk6VDb6pc2ESWt8LdUd3KMGCyYHPC557DNQaXp1kt1cHwnoVzpNmNMmivo30+S18+chfL+VgPNcYfLgNnP3jmgD0XJ9TRk+przSx8E2lvpHgyzOlzPGZfO1LzQ7M7m1dSZyeSM4XDcYwuMYFJ4d8M/2ZdeHbmDSpYJ49SvYpZTE26K2/feXHk/ci4TaowvTAoA9MyfU0ZPqa8z0LRtSh+ITz6gUS9W7nkeaLQ5g9xbtu2K16ZDGyAFcJgEFRhRjNaXhCzk03xjqltb2UptpQ8011PYSW8olZ8hHlJ2XXBO11GUVQCTmgDusn1NGT6mkooAXJ9TRk+ppKKAFyfU0ZPqaSigBcn1NGT6mkooAXJ9TRk+ppKKAFyfU0ZPqaSigBcn1NGT6mkooAXJ9TRk+ppKKAFyfU0ZPqaSigBcn1NGT6mkooAXJ9TRk+ppKKAFyfU0ZPqaSigBcn1NGT6mkooAXJ9TRk+ppKKAB5RHGzyOERQSzMcAD1NZmk+JdN1tnGnzTNsG4Ga2lhEi/wB9C6gSJ0+ZcryOeaj8V6fc6r4Q1WwsNv2m4tZI4t/QsVOAeRwenUVzOn/21qiajY2l7qlxZzafLGzappws/stwRhEixEhZMFsn58YHzHuAdRp3inStWuJYbK5kYxqXEklvJHHKo6tHIyhZFHcoSBkeopdL8T6ZrF29tZS3AmVd4We1lg81M43xmRVEi8j5lyORzyK5fwtd6tDqFvAsurT2dvaMt9bXul/Z47V1ACrbt5amTJDDhpRgA7uRm34X1QeIdcbU7+31O2u1hZLe0udKuLdLaIkEgySRqryNhScHAxgDgswB2WT6mjJ9TSUUALk+poyfU0lFAC5PqaMn1NJRQAuT6mjJ9TSUUALk+poyfU0lFAA8ojjZ5HCIoJZmOAB6msrS/FGl6wJTZXEu2NS+6e3kgV0/voZFUOn+2uV5HPNN8V6fc6r4Q1WwsNv2m4tZI4t/QsVOAeRwenUVh+HL/UxqUzfaNXvtMitWaZNQ0z7K0EwI2xwARIZFxu/v9F+Y55AN/S/E2m6vcSwWctwssa79lxaSwF1zjenmKu9f9pcjkc8imaN4q03X5GXS/tzqqlhLLp9xDE4Bx8sjoFb8CaxvC2pr4g1eTVL631K3vfs5SK0udLuLZLaIkEr5kkah5CQucHHAAGAS0XhW2Fp4jeDQLfV7XR0t3F1Bqfn7I5942iHzs8Y358smPpigDtsn1NGT6mkooAXJ9TRk+ppKKAFyfU0ZPqaSigBcn1NGT6mkooAXJ9TRk+ppKKAFyfU0ZPqaSigBcn1NGT6mkooAXJ9TRk+ppKKAFyfU0ZPqaSigBcn1NZOreJ9M0SZY9QmuAxXe5htZZliX+9IyKRGvB+Z8Dg88GtWuR8X62yXi6JJb6pFZ3EJa6vLPSri7yh48pDFGwViM5LdB0BJyoBs6t4o03RZbaK8e6kkulZoY7OymumdVxk4iRiANw5PHNLqPibT9LhgkuzeHz13pHBYzzSBf7zIiFkAzyWAAPB5rE8XWemG0sZntNbW5hhePT59JW4DxMQMIwi6A7V/1o2fL81M1nXtV07SdJ0++hvo768tgb6/sdMmvFtyFG8KsSMN5JO3I2jk842kA6+3uYru2iuLWZZYZUDxyI2VZSMgg+lSZPqapaRDa22i2cOnxyR2scKrEkqMjhQONysAwPsRmrlAC5PqaMn1NJRQAuT6mjJ9TSUUALk+poyfU0lFAC5PqaMn1NJRQAuT6mjJ9TSUUALk+poyfU0lFAC5PqaMn1NJRQBV1PVbXSLFru/ldIlIUBI2kd2JwFVFBZmPYAEmqkXijSptGk1Rbp1to38txJDIkqvnGwxMocOSQAu3ccjA5FZ/jiOaXSbTy4LloUvI5JrizjeSe1VckSRonzM27AxhhgnKsMisDT7XUBoa3j295dQ2usreGWW1kiur2LbhpHhbDB1J4UKufLG1AMCgDr4vE+nT6RLqUDXksML7JYksZzOjcfKYAnmA8g4K9CD0pdG8S2GvSXMenm8WS1KiZLuxntmUsMjiVFJ49KxbO+ukXxD4itNNvXjnWMWdrLbvHLOyJjcY2UOuWOPmAOFz0xW7oemNpelpHO4mu5T5t1Nj/AFsrfeb6dgOwAHagDVQ5XmnU1Pu/jTqAEbO04644zUO2f+5H/wB9n/Cp6KAINs/9yP8A77P+FG2f+5H/AN9n/Cp6KAINs/8Acj/77P8AhRtn/uR/99n/AAqeigCDbP8A3I/++z/hRtn/ALkf/fZ/wqeigCDbP/cj/wC+z/hRtn/uR/8AfZ/wqeigCDbP/cj/AO+z/hRtn/uR/wDfZ/wqeigCDbP/AHI/++z/AIUbZ/7kf/fZ/wAKnooAg2z/ANyP/vs/4UbZ/wC5H/32f8KnooAg2z/3I/8Avs/4UbZ/7kf/AH2f8KnooAg2z/3I/wDvs/4UbZ/7kf8A32f8KnooAg2z/wByP/vs/wCFG2f+5H/32f8ACp6KAINs/wDcj/77P+FG2f8AuR/99n/Cp6KAINs/9yP/AL7P+FG2f+5H/wB9n/Cp6KAINs/9yP8A77P+FG2f+5H/AN9n/Cp6KAINs/8Acj/77P8AhRtn/uR/99n/AAqeigCDbP8A3I/++z/hRtn/ALkf/fZ/wqeigCDbP/cj/wC+z/hRtn/uR/8AfZ/wqeigCDbP/cj/AO+z/hRtn/uR/wDfZ/wqeigCDbP/AHI/++z/AIUbZ/7kf/fZ/wAKnooAg2z/ANyP/vs/4UbZ/wC5H/32f8KnooAg2z/3I/8Avs/4UbZ/7kf/AH2f8KnooAg2z/3I/wDvs/4UbZ/7kf8A32f8KnooAg2z/wByP/vs/wCFG2f+5H/32f8ACp6KAINs/wDcj/77P+FG2f8AuR/99n/Cp6KAINs/9yP/AL7P+FG2f+5H/wB9n/Cp6KAINs/9yP8A77P+FG2f+5H/AN9n/Cp6KAINs/8Acj/77P8AhRtn/uR/99n/AAqeigCDbP8A3I/++z/hRtn/ALkf/fZ/wqeigCDbP/cj/wC+z/hRtn/uR/8AfZ/wqeigCDbP/cj/AO+z/hRtn/uR/wDfZ/wqeigCDbP/AHI/++z/AIUbZ/7kf/fZ/wAKnooAg2z/ANyP/vs/4UbZ/wC5H/32f8KnooAg2z/3I/8Avs/4UbZ/7kf/AH2f8KnooAg2z/3I/wDvs/4UbZ/7kf8A32f8KnooAg2z/wByP/vs/wCFG2f+5H/32f8ACp6KAINs/wDcj/77P+FG2f8AuR/99n/Cp6KAINs/9yP/AL7P+FG2f+5H/wB9n/Cp6KAINs/9yP8A77P+FG2f+5H/AN9n/Cp6KAINs/8Acj/77P8AhRtn/uR/99n/AAqeigCDbP8A3I/++z/hRtn/ALkf/fZ/wqeigCDbP/cj/wC+z/hRtn/uR/8AfZ/wqeigCDbP/cj/AO+z/hRtn/uR/wDfZ/wqeigCDbP/AHI/++z/AIUbZ/7kf/fZ/wAKnooAg2z/ANyP/vs/4UbZ/wC5H/32f8KnooAg2z/3I/8Avs/4UbZ/7kf/AH2f8KnooAg2z/3I/wDvs/4UbZ/7kf8A32f8KnooAg2z/wByP/vs/wCFG2f+5H/32f8ACp6KAINs/wDcj/77P+FG2f8AuR/99n/Cp6KAINs/9yP/AL7P+FG2f+5H/wB9n/Cp6KAINs/9yP8A77P+FG2f+5H/AN9n/Cp6KAINs/8Acj/77P8AhRtn/uR/99n/AAqeigCDbP8A3I/++z/hRtn/ALkf/fZ/wqeigCDbP/cj/wC+z/hRtn/uR/8AfZ/wqeigCDbP/cj/AO+z/hRtn/uR/wDfZ/wqeigCDbP/AHI/++z/AIUbZ/7kf/fZ/wAKnooAg2z/ANyP/vs/4UbZ/wC5H/32f8KnooAg2z/3I/8Avs/4UbZ/7kf/AH2f8KnooAg2z/3I/wDvs/4UbZ/7kf8A32f8KnooAg2z/wByP/vs/wCFG2f+5H/32f8ACp6KAINs/wDcj/77P+FG2f8AuR/99n/Cp6KAINs/9yP/AL7P+FG2f+5H/wB9n/Cp6KAINs/9yP8A77P+FG2f+5H/AN9n/Cp6KAGRhgvzhQc9jmn0UUAf/9k=)

Next a Correspondence Analysis will be done on this by treating this as a Contingency table.

CA on Bird\_Brain

# Correspondence Analysis

CA is a generalized version of PCA for analyzing data table with qualitative variables. In case of PCA we divided the table into factor scores(rows) and loadings(variables), where as in CA, we derive 2 set of factor scores one for the rows and the other for the columns. -Rows have masses representing each rows importance wrt the total table -similarly Columns have weights each columns importance in describing each of the rows

## Dataset: Bird Brain

The Bird Brain data set contains data about about 28 different birds species, and they are measured using 11 variables, one being categorical and remaining 10 quantitative. One categorical variable groups these birds into different levels. The quant variables describe the count of neurons in various structure of brain.

#Fetching the Bird brian dataset  
  
bb\_df <- read.csv("BirdBrain.csv")  
  
cols <- c('X','Order','Whole.Brain','Pallium','Diencephalon','Cerebellum')  
head(bb\_df[cols])

## X Order Whole.Brain Pallium Diencephalon Cerebellum  
## 1 Starling Passeriformes 691 699 155 852  
## 2 Mynah Passeriformes 704 717 119 876  
## 3 Blackbird Passeriformes 630 612 153 816  
## 4 Goldcrest Passeriformes 788 803 400 883  
## 5 Zebrafinch Passeriformes 695 703 305 840  
## 6 Tit Passeriformes 662 625 252 807

#### **Working on dataset**

row.names(bb\_df) <- bb\_df$X  
new\_bb\_df <- bb\_df[-c(1,2)]  
B <- as.matrix(new\_bb\_df)

### **Heatmap**

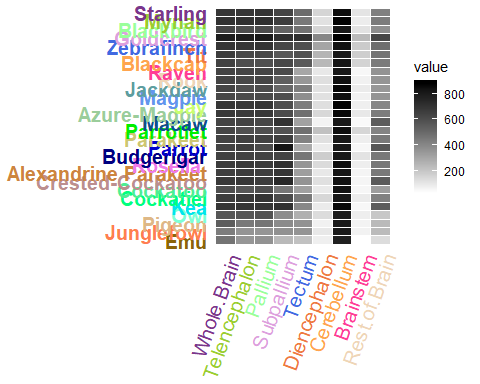
**Heat map shows that the cerebellum part is the most crowdedwith neurons for all the species observed here.**

-Goldcrest has more density in 3 more portions of brain.

-Parrot has a rare higher density at subpallium

-While EMu has pretty musch lesser neurons than rest in all regions except cerebellum.

col4J.Bird <- prettyGraphsColorSelection(NCOL(bb\_df))  
c000.heatMapIJ.IBM <- makeggHeatMap4CT(new\_bb\_df,  
 colorAttributes = col4J.Bird,  
fontSize.x = 15)  
print(c000.heatMapIJ.IBM)



### **1st Design Variable : Order**

Selecting order as the grouping variable

#Create an empty list, called DESIGN  
DESIGN <- list()  
  
DESIGN$rows$Order$labels <- unique(bb\_df$Order)  
DESIGN$rows$Order$vec <- bb\_df$Order  
DESIGN$rows$Order$mat <- makeNominalData(as.matrix(DESIGN$rows$Order$vec))

### **Assiging color to the groups**

DESIGN$rows$Order$color\_groups <- c("red", "green", "magenta", "blue","yellow","brown","orange")  
  
DESIGN$rows$Order$color\_observ <- as.matrix(DESIGN$rows$Order$vec)  
count = 0  
for (val in unique(bb\_df$Order)) {  
 count = count + 1  
 DESIGN$rows$Order$color\_observ[which(DESIGN$rows$Order$vec==val)] <- DESIGN$rows$Order$color\_groups[count]  
}

## Running CA

Running the symmetric and asymmetric as well as the inference pca

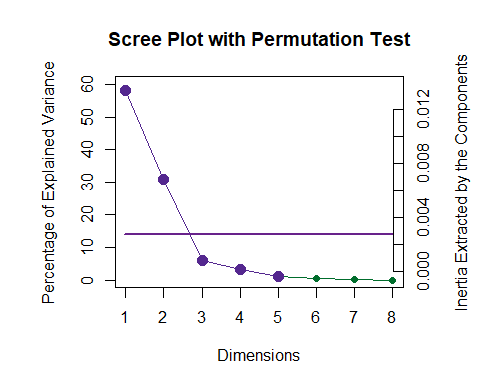
# Running the Symmetric CA  
resCA.sym <- epCA(B, symmetric = TRUE,DESIGN = bb\_df$Order)  
  
# to run a plain CA but asymetric  
resCA.asym <- epCA(B, symmetric = FALSE,DESIGN = bb\_df$Order)  
  
  
resCA.inf <- InPosition::epCA.inference.battery(DATA = B,  
 #scale = 'SS1', # Make sure to use 'SS1' rather than TRUE  
 DESIGN = bb\_df$Order,  
 graphs = FALSE # TRUE first pass only  
 )

## [1] "It is estimated that your iterations will take 1.73 minutes."  
## [1] "R is not in interactive() mode. Resample-based tests will be conducted. Please take note of the progress bar."  
## ===========================================================================

### **Scree Plot**

**From the permutation test on the scree plot, we can see that 2 components are of significance**

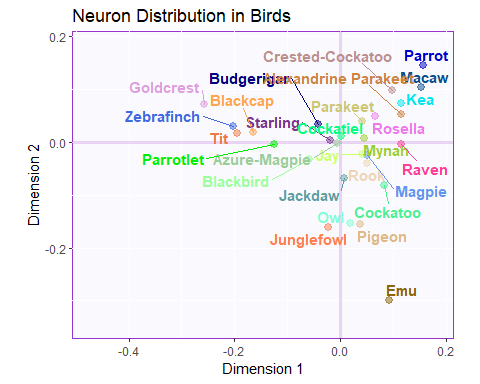
EigenValues <- resCA.asym$ExPosition.Data$eigs  
  
  
PlotScree(ev = EigenValues,   
 p.ev = resCA.inf$Inference.Data$components$p.vals,  
 title = "Scree Plot with Permutation Test",  
 plotKaiser = TRUE  
 )



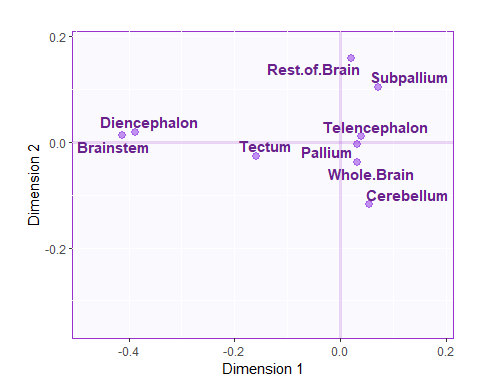
a001a.screePlot <- recordPlot()  
print(a001a.screePlot)

### **Creating Base maps**

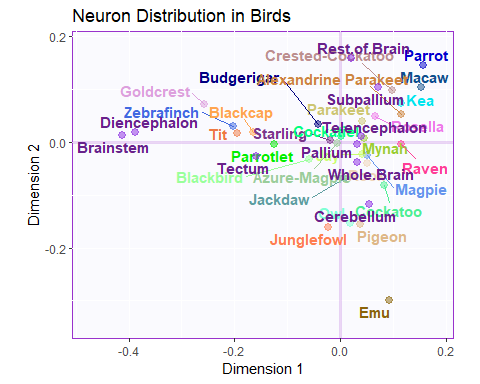
# Get some colors ----  
color4Authors <-prettyGraphsColorSelection(n.colors = nrow(Fi))  
# baseMaps ----  
baseMap.i <- createFactorMap(Fi, constraints = constraints.sym,  
 col.points = color4Authors,  
 col.labels = color4Authors,title = "Neuron Distribution in Birds")  
  
print(baseMap.i$zeMap)



baseMap.j <- createFactorMap(Fj, constraints = constraints.sym,  
 color.points = 'darkorchid4')  
  
print(baseMap.j$zeMap)



print(baseMap.i$zeMap + baseMap.j$zeMap\_dots + baseMap.j$zeMap\_text)



### **Plotting observation for symmetric and asymmetric**

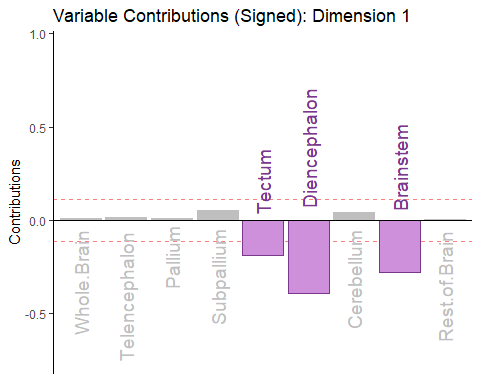
symMap <- createFactorMapIJ(Fi,Fj,  
 col.points.i = DESIGN$rows$Order$color\_observ,  
 col.labels.i = DESIGN$rows$Order$color\_observ)  
  
asymMap <- createFactorMapIJ(Fi,Fj.a,  
 col.points.i = DESIGN$rows$Order$color\_observ,  
 col.labels.i = DESIGN$rows$Order$color\_observ)  
  
labels4CA <- createxyLabels(resCA = resCA.sym)

## Inferences

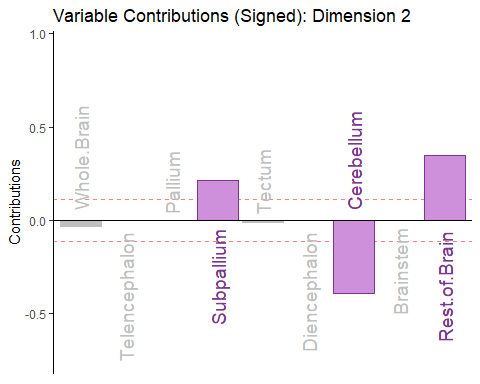
### **Contribution Plot**

**Tectum,Diencephalon and Brainstem have significant contributions to the Dimension 1** **Whereas, Subpallium,Cerebellum and Rest of Brain have more significant contributions in Dimension 2**

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Contribution Plots ----  
  
signed.ctrJ <- resCA.sym$ExPosition.Data$cj \* sign(resCA.sym$ExPosition.Data$fj)  
  
b003.ctrJ.s.1 <- PrettyBarPlot2(signed.ctrJ[,1],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(resCA.sym$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 1',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
print(b003.ctrJ.s.1)

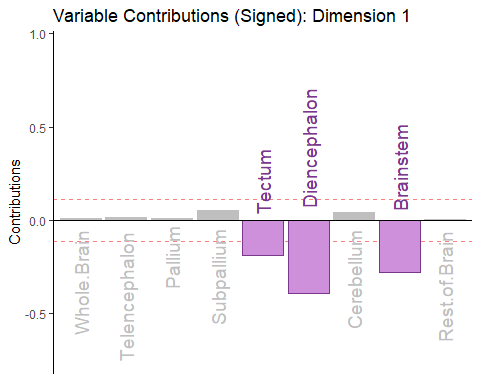


b004.ctrJ.s.2 <- PrettyBarPlot2(signed.ctrJ[,2],  
 threshold = 1 / NROW(signed.ctrJ),  
 font.size = 5,  
 color4bar = gplots::col2hex(resCA.sym$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 2',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ), 1.2\*max(signed.ctrJ))  
)  
print(b004.ctrJ.s.2)

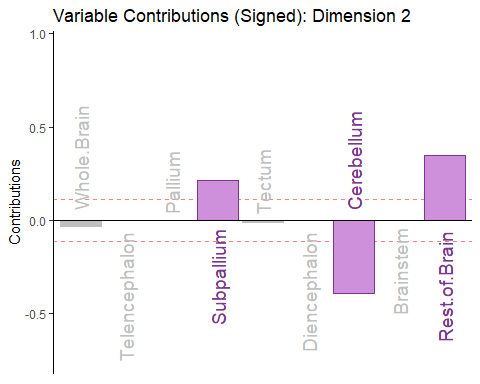


### **Contribution Plots Asymmetric**

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
# Contribution Plots ----  
  
signed.ctrJ.a <- resCA.asym$ExPosition.Data$cj \* sign(resCA.asym$ExPosition.Data$fj)  
  
b003.ctrJ.a.1 <- PrettyBarPlot2(signed.ctrJ.a[,1],  
 threshold = 1 / NROW(signed.ctrJ.a),  
 font.size = 5,  
 color4bar = gplots::col2hex(resCA.asym$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 1',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ.a), 1.2\*max(signed.ctrJ.a))  
)  
print(b003.ctrJ.a.1)



b004.ctrJ.a.2 <- PrettyBarPlot2(signed.ctrJ.a[,2],  
 threshold = 1 / NROW(signed.ctrJ.a),  
 font.size = 5,  
 color4bar = gplots::col2hex(resCA.asym$Plotting.Data$fj.col), # we need hex code  
 main = 'Variable Contributions (Signed): Dimension 2',  
 ylab = 'Contributions',  
 ylim = c(1.2\*min(signed.ctrJ.a), 1.2\*max(signed.ctrJ.a))  
)  
  
print(b004.ctrJ.a.2)

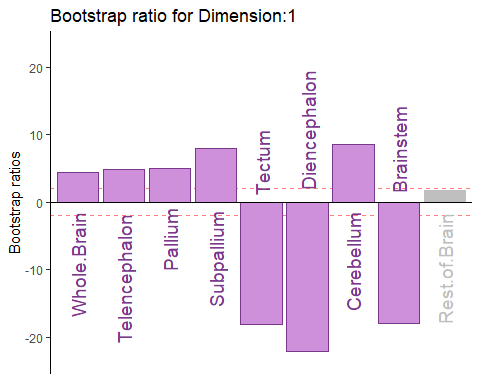


### **Bootstrap Bars**

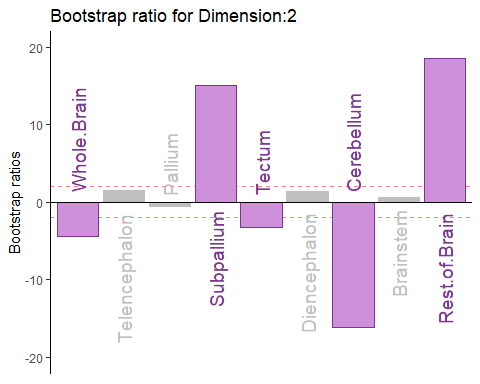
For Dimension 1 its shows, all the variables have significant contribution

For Dimension 2 its shows, Subpallium,Tectum,Diencephalon and Brainstem have significant contribution

BR <- resCA.inf$Inference.Data$fj.boots$tests$boot.ratios  
laDim = 1  
ba001.BR1 <- PrettyBarPlot2(BR[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resCA.sym$Plotting.Data$fj.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios')  
  
print(ba001.BR1)



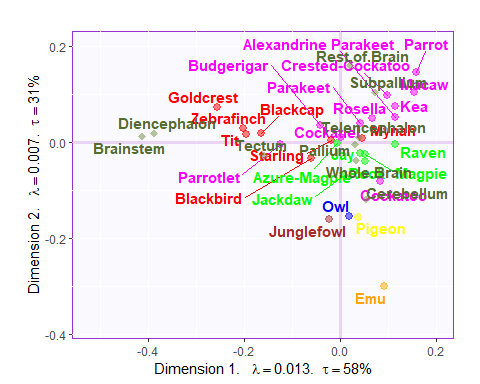
#  
laDim = 2  
ba002.BR2 <- PrettyBarPlot2(BR[,laDim],  
 threshold = 2,  
 font.size = 5,  
 color4bar = gplots::col2hex(resCA.sym$Plotting.Data$fj.col), # we need hex code  
 main = paste0(  
 'Bootstrap ratio for Dimension:',laDim),  
 ylab = 'Bootstrap ratios'  
)  
  
print(ba002.BR2)



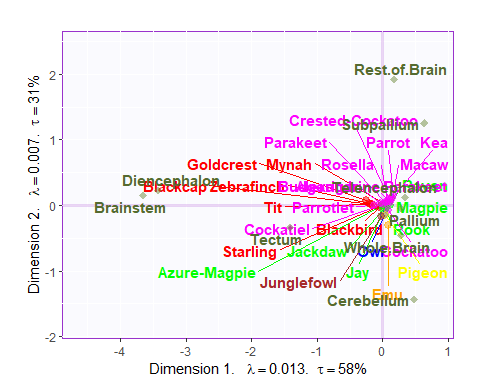
### **Symmetric and Asymmetric Plots**

**From the symmetric plot:** - we can interpret that the Passeriformes(GOldcrest, STraling etc) have more neurons towards Brainstem & Diencephalon compared to others. - Psittaciformes(Parrot, Parakeet) have more neuron towards Subpallium region - Corvid(Raven, MAgpie) are towards Pallium - WHile Emu, Pigeon, Owl etc ae far from others but still closer to the cerebellum region

# draw the maps ----  
  
  
map.IJ.sym <- symMap$baseMap + symMap$I\_labels + symMap$I\_points +  
 symMap$J\_labels + symMap$J\_points + labels4CA  
  
print(map.IJ.sym)



map.IJ.asym <- asymMap$baseMap + asymMap$I\_labels +   
 asymMap$I\_points + asymMap$J\_labels +   
 asymMap$J\_points + labels4CA  
  
  
print(map.IJ.asym)



## Conclusion

### **Component 1:**

**Rows:** Passeriformes VS Psittaciformes & Corvid

**Columns:** Brainstem,Diencephalon and Tectum VS Others

**Component 2:**  
**Rows:** Struthioniformes(Emu) & Columbiformes(Pigeon) & Galliformes(Junglefowl) VS Rest **Columns :** Rest of Brain & Subpallium VS Cerebellum & Whole Brain

**Interpretation:** Majority birds are concentrated near the whole brain, palium,telencephalon region. Few birds of Passeriformes order have neurons in Diencephalon & Brainstem. While EMu shows more neurons near the Cerebellum area.

## Data set: Wine Data

We have the data related 18 wines, that have been assessed by 19 different experts.

Wines tested basically are of 3 colors (Red, White, Rose).

TO find a general trend amongst the experts I randomly assign them gender (this was not given in data, any conclusion from this will not be significant).

(**ROWS**)**: 18**

(**COLUMNS**): **19**

A head command shows the following snap:

![A picture containing sky

Description generated with high confidence](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAkACQAAD/4RDcRXhpZgAATU0AKgAAAAgABAE7AAIAAAAGAAAISodpAAQAAAABAAAIUJydAAEAAAAMAAAQyOocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGFqYXljAAAFkAMAAgAAABQAABCekAQAAgAAABQAABCykpEAAgAAAAM2MQAAkpIAAgAAAAM2MQAA6hwABwAACAwAAAiSAAAAABzqAAAACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMjAxODoxMToyMSAyMTo1MTowMQAyMDE4OjExOjIxIDIxOjUxOjAxAAAAYQBqAGEAeQBjAAAA/+ELGGh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8APD94cGFja2V0IGJlZ2luPSfvu78nIGlkPSdXNU0wTXBDZWhpSHpyZVN6TlRjemtjOWQnPz4NCjx4OnhtcG1ldGEgeG1sbnM6eD0iYWRvYmU6bnM6bWV0YS8iPjxyZGY6UkRGIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iLz48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOnhtcD0iaHR0cDovL25zLmFkb2JlLmNvbS94YXAvMS4wLyI+PHhtcDpDcmVhdGVEYXRlPjIwMTgtMTEtMjFUMjE6NTE6MDEuNjEyPC94bXA6Q3JlYXRlRGF0ZT48L3JkZjpEZXNjcmlwdGlvbj48cmRmOkRlc2NyaXB0aW9uIHJkZjphYm91dD0idXVpZDpmYWY1YmRkNS1iYTNkLTExZGEtYWQzMS1kMzNkNzUxODJmMWIiIHhtbG5zOmRjPSJodHRwOi8vcHVybC5vcmcvZGMvZWxlbWVudHMvMS4xLyI+PGRjOmNyZWF0b3I+PHJkZjpTZXEgeG1sbnM6cmRmPSJodHRwOi8vd3d3LnczLm9yZy8xOTk5LzAyLzIyLXJkZi1zeW50YXgtbnMjIj48cmRmOmxpPmFqYXljPC9yZGY6bGk+PC9yZGY6U2VxPg0KCQkJPC9kYzpjcmVhdG9yPjwvcmRmOkRlc2NyaXB0aW9uPjwvcmRmOlJERj48L3g6eG1wbWV0YT4NCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgPD94cGFja2V0IGVuZD0ndyc/Pv/bAEMABwUFBgUEBwYFBggHBwgKEQsKCQkKFQ8QDBEYFRoZGBUYFxseJyEbHSUdFxgiLiIlKCkrLCsaIC8zLyoyJyorKv/bAEMBBwgICgkKFAsLFCocGBwqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKv/AABEIAMED+QMBIgACEQEDEQH/xAAfAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJCgv/xAC1EAACAQMDAgQDBQUEBAAAAX0BAgMABBEFEiExQQYTUWEHInEUMoGRoQgjQrHBFVLR8CQzYnKCCQoWFxgZGiUmJygpKjQ1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6wsPExcbHyMnK0tPU1dbX2Nna4eLj5OXm5+jp6vHy8/T19vf4+fr/xAAfAQADAQEBAQEBAQEBAAAAAAAAAQIDBAUGBwgJCgv/xAC1EQACAQIEBAMEBwUEBAABAncAAQIDEQQFITEGEkFRB2FxEyIygQgUQpGhscEJIzNS8BVictEKFiQ04SXxFxgZGiYnKCkqNTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqCg4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2dri4+Tl5ufo6ery8/T19vf4+fr/2gAMAwEAAhEDEQA/APpGiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAbI4jQs3SmR3CSPtGQe2e9QX0UsrIEXKgHLen4VIoBePap+U8naR2NAFiiioLyMzWckYj80sMbC5UN9SO3r7UAOguYbqMvbSpKoYqShyMipaoWVnNbXTtPskLIAJUGzp/CVHHHY9cce5v0AIWC43EDJwMnqaAwLEAgkdRnpVPV7WS9017eGON3kIAMhIEfP3+OcjqMY57iotGsJtPhmiuCsrmQt9pz80+e7ehHT04GMdAAaVFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFNkkSJN8rqi5A3McDngUAZUXirRJY9Vk/tGGKLR5jBfyz5iSBwoYgswA6MOQSOat6Rq1jruk2+p6TOLmzuV3wyhSA4zjOCAe1cP4Fid774hui71m1uYRledxEEYIGPcYrV+E9ldad8J/D1pqFtNaXMNoFkgnjKOhyeCp5BoA6+iiigAooooAKKKKACiiigArPv9c07TNR06xvrjyrnU5WhtE2M3mOqliMgYHygnnFaFcB42K3PxE8AC3ZZDFqNyzhWBKgW7A/kSBQB0MPjXQptah0sXU0dzcMy27TWk0cNwQMkRysgjkOOcKx7+lb1ebWt+PGPjqyudcsdY0+30q6c6ZYyaPdKJJQCv2iacx+WBgttUNjBySSdq+k0AFFFFABRRRQAUUUUAFFFFAEdzcRWlrLc3DbIoUMjtgnCgZJwPasSXxvoMWh2mrrc3E9leQi4ha2sZ53MWM+YURCyrjqWAAyAateJ54ovC+qCWVEJsZyAzAZAQ5NcFpLz2fwV8MWkk+u2LNpMYLaTp/2syN5Q/duFjkZQc/eGz/fBxQB6Vp+oWmq6db3+nXEdzaXMYkhmjOVdSMgirFYXgi2ubPwLo9vfabFpdxFaIj2UOdsJA+6Mlj+ZP1PWt2gAooooAKKKKACiiigAoopHdY0Z5GCqoyzMcAD1oAoW+uadda9eaNBcbtQsoo5riHYw2JJnYdxGDnaehPSo9M8SaRrGqajp2mXyXN1pjrHdogOImIyBnGCeOxOO9cNFY32qfFTxzHo2oLYTTaXYQx3oj80RNiQn5Qy87T6jqDV3wPoOtaJ421qK+azNglnaxQva6dJbpKQHPys0r5Iyd3JJJB47gHoFFFFABRRRQAUUUUAFFFFABWfrGuadoFtDcatcfZ4p7iO2jbYzbpJG2ovyg9SevT1rQrgPi0RcaHo1vCyvN/wkFh+7DDdxKG6fQZoA6O48X6Tba3JpJ+3zXkRQSLbaZczpGX5XdIkZReOeSMd63K858Z2lxB4iguPCcGuW/iK4vLcu8InaxnhBAcy5JgACbhziTIGB0z6NQAUUUUAFFFFABRRRQAUUUUAFYC+OPDzaCNZS/L2TXDWqMkEjPJMrlCiRhd7tuBGFBzjI4rdeWOLb5jqm47V3HGT6fpXhPh3Sb8+FfDGsQm8Szs9X1N5prC0F1NCsskgSaOMK+7njIR8BskYyQAe3adqVtqtmLmzZyhOCssTROh/usjgMp5HDAHmrVcz4Eu9cvNEnk8Q+e+LqRbOa6txBPNbg/I8sYC7GPPG1eAOBXTUAFFFFABRRRQAUUUUAFFFFAGfLrunQ+IIdDkuMajPbvdRweWx3RqQrNuxjgsOM5rPg8b6NcavbaZjU4bq6dkgFzpF3AsjKCxAd4gvQE9awLoi5+PukTW7LLGnh+43MjAgbpkx+eD+Va/hpLvVtc1PXtUtprbbK9lp8FxEUaOBDhnwef3jDOe6qtAHUUUUUAFFFFABRRRQAUUUUAFUtZ1iw8P6Nc6rq8/2eytE8yaXYz7F9cKCT+Aq7XD/GJ0f4SeI7dXTzns8KhYAncwA/M8UAbWreNdC0SZI7+6mDND9obyLOacRRdnkMaMI193wOD6GtyKVJoUlhdZI3UMjqchgehBrxzxppuqT69dmO01OyL6PDb20thYz3UWp/e329z5ZwgGcAnyyNzHeRkV6zo4uF0OxW9to7S4Fuglt4mykTbRlVPoOlAFyiiigAooooAKKKKACiiigArBm8a+Hre21OebUQsel3ItLrMT5WYgERqu3Lsdy4Cg5JwOa3JJEiTfK6ouQNzHA54FeS6Vo1lq6+O21JL4wR+JvtMFxpql5oXjjjxIigMWKkH5drZ6bTQB6Xo+vafrsUradJKWgfZNDcW8kEsRxkbo5FV1yORkcjpWjXB+AtLvj4o17X7qTVHtr1ILe3fVIhDNcCINmUxBU8vJbaAUU4XOOcnvKACivLvAXxcj1/4heIPBesR+XqNhqN3HZSxp8s8McrAK2OjKo69CB69fUaACiiigAooooAKKj+0wf89o/++hR9pg/57R/99CgCSio/tMH/AD2j/wC+hR9pg/57R/8AfQoAkoqP7TB/z2j/AO+hR9pg/wCe0f8A30KAJKKj+0wf89o/++hR9pg/57R/99CgCSio/tMH/PaP/voUfaYP+e0f/fQoAkoqP7TB/wA9o/8AvoUfaYP+e0f/AH0KAJKKj+0wf89o/wDvoUfaYP8AntH/AN9CgCSio/tMH/PaP/voUfaYP+e0f/fQoAkoqP7TB/z2j/76FH2mD/ntH/30KAJKKj+0wf8APaP/AL6FH2mD/ntH/wB9CgCSio/tMH/PaP8A76FH2mD/AJ7R/wDfQoAkoqP7TB/z2j/76FH2mD/ntH/30KAJKKj+0wf89o/++hR9pg/57R/99CgChdalNFfJblY7OMsAJ7kErJyPlXBwCc4G5gcj7pFVZrm5uoGtY0tmSVbnzPtKNKpVX24255BBxjIH8q2ftMH/AD2j/wC+hUYNkDkGAHnpt7nJ/M9aAMLTQNOjWDSv7O0jS47FZ0t0tAkcLMSSflZQBnr0/DrWxp17LeIxltygU/LKM7JOSMjcA3bPTHPBPWn7NP8AMik222+FdsTYXKD0B7DgdKm+0wf89o/++hQBJRUf2mD/AJ7R/wDfQo+0wf8APaP/AL6FAElFR/aYP+e0f/fQo+0wf89o/wDvoUASUVH9pg/57R/99Cj7TB/z2j/76FAElVdQu5bO2MkFq9w3PC5wvucAt7fKrH2qb7TB/wA9o/8AvoUfaYP+e0f/AH0KAMiTU5Uh86Oe3umW3nkDxbgmVK4UjcenQ9+O3SqFzbmC+vdQhtdMg1jECm+SyBkMTtjy2bO5gMdcgewro5Gs5f8AWtA/ylfmIPB6j6HFNkFjKsiyi3dZV2SBtpDr6H1HNAFRdRuI9RNr+6vucP8AZlKtCeOGySvcnlgcdA3WtSoUlto41SOSJEUYCqwAAp32mD/ntH/30KAJKKj+0wf89o/++hR9pg/57R/99CgCSio/tMH/AD2j/wC+hR9pg/57R/8AfQoAkoqP7TB/z2j/AO+hR9pg/wCe0f8A30KAHscKSAWIHQd6xZdWneO7EgjtWjhdltZNwmbAB3bgQMDODs3D/a7Vr/aYP+e0f/fQprzW0iMkkkTKwwVZgQRQBh6rFFqzG21bT9OvtPiu4ozBdW/nbiQuGG7gEFuuD+HWnxX99bWcss93ZsVuZI4YTAyGQKThF2liTx2UnAPB61rk2ZzkwHLBz05YdD9eB+VJH9iimkli+zpJKcyOu0Fz05PegCS2meeASSQPAx/gcjP14Pf8/UDpUtR/aYP+e0f/AH0KPtMH/PaP/voUASUVH9pg/wCe0f8A30KPtMH/AD2j/wC+hQBJRUf2mD/ntH/30KPtMH/PaP8A76FAElFR/aYP+e0f/fQo+0wf89o/++hQBR1PUZrRwiokEZHN5cAmJOCeQv0/iKDngk8VE97c/aGhtvJLSXQi3ybioHlbs4z69hgH9a0/tMH/AD2j/wC+hUe6z378wbt27dkZzjGfrjigDn9Jt10+S0i0e303TIJzPJdx21kI1lkVgC/BGCffPvmtfTNRlvGKMiTxgcXlvkRScDoG9c/wlxxy2eKmaPT3EYdLVhG+9AQp2N6j0PJ596n+0wf89o/++hQBJRUf2mD/AJ7R/wDfQo+0wf8APaP/AL6FAElFR/aYP+e0f/fQo+0wf89o/wDvoUASUVH9pg/57R/99Cj7TB/z2j/76FAElQ3Uz29uZIoHnYfwIRn689vpk+gPSnfaYP8AntH/AN9Cj7TB/wA9o/8AvoUAZcGpSXH2Zmlgdnn2tFFvUw/uy21skEn/AHlXr90HmsudPtUiatdWWmHU49Pea0u/sgeW39QHY5wd3QYx710zS2zlS7wsVOVJIODjHH500GyXbtMA2psXG3hfT6cdKAM/+0bu2kgjd4L6Z41LwQIUkXIJ3YyQAcY+YqM5+boK2AcgEjHse1V4DZWsKw2xghiXhUj2qo+gFSfaYP8AntH/AN9CgCSio/tMH/PaP/voUfaYP+e0f/fQoAkoqP7TB/z2j/76FH2mD/ntH/30KAJKKj+0wf8APaP/AL6FH2mD/ntH/wB9CgCSsv8AtOZtSFvIsdkuflWcEvPweEIIXtngscdQtaH2mD/ntH/30KPtEH/PaP8A76FAHOX0zahozwXlpY3Vl9jimngu4jMsoJOVwxwenU5x6GrMEk2nm8hiubCzsbR44raE2+xI12rhBhgOc4GB19elauLHYUxb7SoQr8uCo6D6c9KB9hFybgfZxOV2GX5dxX0z1xQAmn3Ut5aiSe1e3bjhujcdRkBsdvmVT7VaqP7TB/z2j/76FH2mD/ntH/30KAJKKj+0wf8APaP/AL6FH2mD/ntH/wB9CgCSio/tMH/PaP8A76FH2mD/AJ7R/wDfQoAkoqP7TB/z2j/76FH2mD/ntH/30KAK2o301mq+VblwxAaZslI+cZIUFvfoBxywqk+pTqGe3kt7ljHb7ZFLCN97lSQNxwPpn8a1vtMH/PaP/voVGzWbsWcwMxxkkg5wcj8jzQBz0FqbK9eXTbfTLO+ub/y725gsQpuvkLAthsk89ST/AErWtNRmlvXtwI7uNWINxbghU5PytnjIxg7WJyeVUVZlTT5o2SZbaRHYOyuFIZh0JHrwOfapRcQAYE0YH+8KAJaKj+0wf89o/wDvoUfaYP8AntH/AN9CgCSio/tMH/PaP/voUfaYP+e0f/fQoAkoqP7TB/z2j/76FH2mD/ntH/30KAJKZM7RQs6RPMyjIjQgM3sMkD8zSfaYP+e0f/fQo+0wf89o/wDvoUAZI1SWaNy7wo6yxL9mXeskYMm3cxO04I6fKB15YVVuII9W1Czk1Ww025EVxMto01qJXgkTOJFZj8p+XoBnvntW88ttIuJJImGQcMwPI5BpoazBUgwAqxZcY4J6ke5yfzoAyLXU76LTrKW5ubW6uJ1z5CxMjyDIGV2lumeeMdMlRzW5E7SQo7xtEzDJjcjK+xwSPyJqGH7FbBhb/Z4g7FmEe1dzHqTjvUv2mD/ntH/30KAJKKj+0wf89o/++hR9pg/57R/99CgCSio/tMH/AD2j/wC+hR9pg/57R/8AfQoAkoqP7TB/z2j/AO+hR9pg/wCe0f8A30KAJKzLrUpor5LcrHZxlgBPcglZOR8q4OATnA3MDkfdIq/9pg/57R/99Cj7TB/z2j/76FAGNNc3N1A1rGlsySrc+Z9pRpVKq+3G3PIIOMZA/lVfTQNOjWDSv7O0jS47FZ0t0tAkcLMSSflZQBnr0/DrW6DZA5BgB56be5yfzPWm7NP8yKTbbb4V2xNhcoPQHsOB0oAZp17LeIxltygU/LKM7JOSMjcA3bPTHPBPWrtR/aYP+e0f/fQo+0wf89o/++hQBznhXwFo/hPU9Z1Oyj83UdZvZrq5upB82HkLiNfRRn8SMntjp6j+0wf89o/++hTkkSTPlurY67TnFADqKKKACiiigCpquoxaRo93qNyrtDaQvM6xgFiqjJxkgZ49aztE8TrrF7JZ3GlX+lXSwLcJFe+UTJExwGUxSOvUYIJB6cVN4qsrjUvCGr2VlH5lxcWcsUSbgNzMhAGTwOfWuf0zwvfaPJfWFqs8ttqdhsW/nuPOms5AhXy2d2LumSWXBbaSw4BFAHTad4g0bV7m5t9J1exvp7U7biK2uUkaE5IwwUkryCOfQ1JZavpupWkl1p2oWt3bxMyyTQTq6IV+8CwOAR39K5LTNL1K8u9KiufDn9lQ6PZy2zvJPCY7rcgTZF5ZZhGSu4lgh4X5Sc4qW3hnXLvwvr2jiO7sreSKOHT/ALe9s8wVM/u98YYNFjCqZAz8tuB6UAdVc+LNITwvc6/p91Hq1jbqSW06aOXeQcFVO4LnJ7kVb0nUbnUrZ5bvR73SmVsCK8eFmcY+8PKkcY7ckHjpXNaFo90NF1eXWNM1LUZL7Z51lrT2RacKu0qFgXysYwBuOSRg7QAaveC9Jm0uHUCNN/saxuLgSWell0P2VdoDcRlkTcwLbUJHOepIoA6aiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooApatqtvo2nPeXe8orKipGu55HYgKijuSSAPrSaVqMupWzPcabeaZKjbWt7wJuHoQ0bMjA+zHHQ4PFZ/i/SbjVtIgFn5rS2l1HciOGQRvKFPzIrHhWKk4ORzjkdRB4PsdQtEvWuotQtLKV1NrZ6ne/a7iHAw5aTzJOCeQN7Y9s4AB0tFFFABRRRQAUUUUAFFFFABRRRQBm2mtLda9faS1lc28toiSiSXyykyOWAZNrE9VIwwU8dK0qx7exuU8aX1+0eLaWxgiR9w5dXkJGOvRh+dbFABRRRQAUUUUAFFFFABRRRQAVk6xr6aVPBawWF3qd9cKzx2dn5YcouNzkyOiADIHLDJIAzWtXHeLNEuJ9ch1SKz1XUIDam2lttJvxaTA7wyvvMkZ2/eBAcduG7AHWW04urWOdY5IxIobZKhRl9iD0NS1m+HrfULTw/aQaxN594iYkctuPU4Bb+IgYBPcjNaVABRRRQAUUUUAFFFFABRRRQAyeUQW8krK7iNSxWNSzHHYAdTWXoviFdXubq0m0680y9tQjS2t55ZYI+drAxu6kHa38WeOQK0rrzDZyiGPzJCh2p5hj3HHTcOR9a5jwbp17ZXmozSaffabYzCPyoNTukurlpADudpQ8hK42gBpGIwcBRgUAdZRRRQAUUUUAFFFFABRRRQAUUUUAYmoeJk0zVorW60vUBaySpB/aIWPyFlc4VMF/MOSQNwQrk9eDjbrjtd03ULzxXZzWOl3wlinicX7XyvZJED84MDvxIRuAZYs8j5xzjsaACiiigAooooAKKKKACiiigAqlqmoS6daCW30671KVmCrb2gTe3vmRlQADuWHoMkgVdrH8TQvPo5jXT9Qv0LjfDpt+bS4x6q/mR9+oLrxnr0IBa0bV7fXNKiv7RZERyytHKuHjdWKspwSMhgRwSOOCavVjeErO9sPDNtbalGsMqF9sI2ZiQsSiMUAUsFIBI4Jzyep2aACiiigAooooAKKKKACiiigBGYKpY5wBk4GT+VY2m+JBfap9gutK1DTJnjaW3+2LHi4RSAWXY7Fcbl+Vwrc9ODjYlZkidkQyMqkqgIG4+mTXJ+FG1W71SW/wDEegalZahJEVEtxNbNBAmQfJiEczt6EsVBYjkgBVAB11FFFABRRRQAUUUUAFFFFABRRRQBm3+tLp2r6dYy2Vy66g7RJcx+X5cbhS21gWDchTghSOOSK0qx9bsbm81LQ5bePelrfGWY7gNqeTIueevLAcetbFABRRRQAUUUUAFFFFABRRRQAVW1C8awsnnjtLi8cYCW9soLyE9ANxCj6sQB3IqzWV4ku9Vs9Dml0DT21C+JVY4kZAVBOC/zuinaMnaWGcYyKAH6NrKaxDOfstxZXFtL5Vxa3ITzImwGAJRmU5VgcqxHPrmtKsPwpA0GlP52m6jYzvKXmbUpYZJrhyBmQmJ3XnoACAAAAAABW5QAUUUUAFFFFABRRRQAzefajefasDxt/wAiFrv/AGD5/wD0A1yXg+3XQ7jWJYNFstHvYNNjddK00ZhugAWWcMFTcxPyH5ARjBJypoA9M3n2qI3kIuhbGaPzyhkEW4bioON2OuMnGa830vxT4jtvDmo6pqCXjxHTftlvLqBs9ok7+Wtu5JiAYH5/mGOWOeJ5ft+neLLnytZl1i9j8PTTRGaOISKxcFRiNVBUkccZ4PJ7AHo28+1G8+1eeeDlt9N1bT1bQrO1m1azMyajbXnmz3e0KzNcjYuW+bIbdIASQCM8+gUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefaszXreK70We2mt7W7EwCLbXkxihnYnhHYK3BPBG1s9MGsTwDBHZ2WqWUdvHZtb37K9nbEG2tiUQ7IcY+TBB5VTuLfKvSgDrt59qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2rzc6jqs3xF0STVfDmp280k1ykTvLatFHEEx8u2Yt6MxKgngAHAFei0AP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59qZXM+L7TSL/7Na3+jW+t6hKGFlZ3ChkU8ZlO7IQLxl8bhnAySAQDqd59qN59qoaRZy6fotnZ3Nw1zNbwJG8zEkyEDBPPNXKAH7z7Ubz7UyigB+8+1G8+1MooAfvPtRvPtTKKAH7z7Ubz7UyigB+8+1G8+1RuiyRskih0YEMrDII9DXOeBLaCy8PTW1nBHb28OoXaRxRIFVFE7gAAcAD0oA6fefajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59q4rVNI07VvFiHSdPhOq2txFLeawR+8tlBDeSr/eJZePLBChWJPUBuyoAfvPtRvPtTKKAH7z7Ubz7UyigB+8+1G8+1MooAfvPtRvPtTKKAH7z7Ubz7Uyuf8eWdrfeA9YjvbaG4RLSSRVmjDhXVSVYA9CDyD2oA6Pefajefaq9p/wAeUH/XNf5VLQA/efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2qNiApLHAA5OcYrhfDlhb6Z49by9NsdMN5YySxDTJvNS8USL++nYqjGT5hjIb7zfOc4oA77efajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plFAD959qN59q4vxdpOm63qAsrfTYLzXWiBivJRk6Ymf8AXBjkxnPQJgsR6AsvYICqKpJYgYJPegCTefajefamUUAP3n2o3n2plFAD959qN59qZRQA/efajefamUUAP3n2o3n2plZniH+y/wCw5v7ejWax4EkLKWExJwE2D7+44G3nOcYNAGtvPtRvPtXO+ENIfSNOuB9ij02C4nM0GmwkbLNNoAQBflBJBYheAWOCep36AH7z7Uqtuzmo6fH3oAfRRRQAUUUUAV57WO5t5ILmFJoZFKPHIoZXU8EEHgio2061a6guWtIDPbqUhlMa7olPUKeoBwOB6VF4gvrrTPD19fWESTT20JlWN84YLyRwRzgHHvWC/jcx+LJLR4oRpC6cblbrd8xlVRIU64I8tlNAG5ZaBpWmSXT6dpVlaPeMXuWgt0jM7HOS+B8x5PJ9aTTfDukaKoGj6RY2AUEAWtskWATkj5QOpAP4Vy6eNNT03Vr5dXsLu8W20y1u5obCFAttu8wytudhnG0fLuZuOFPNauseLjpZ1SS3tZL8WOmJfrbxqsbOGLj/AFjPjonTaMY6sTgAGnp/h7SdJurm50vSbGyuLtt1xLbW6RvMck5cqAWOSTz6mtDYfSufuPGP2W1tnl0LVGuZYDczWkfkNJawg4MjnzNpHoFZmPYcHGd4t8Vajpd9pzafd21lpNxbPPNqdxpFxewxnKbdzROixKVZiWc4wO1AHY7D6UbD6UsRLQoS6uSoJdBhW9xyePxNPoAj2H0o2H0qSigCPYfSjYfSpKKAI9h9KNh9KkooAj2H0o2H0qSigCPYfSjYfSpKKAI9h9KNh9KkooAj2H0o2H0qSigCPYfSjYfSpKKAI9h9KNh9KkooAj2H0o2H0qSigCrdWUF9aS2t9bxXNvMpSWGZA6Op6gqeCPY0yw0uz0qySz0uyt7K1jzsgtoljRc88KuAKkv76LTrNrifcwBCoiDLSMeFVR3JPArP8N6pfanBff2nHDFNbXjwBYc4CgKQCSeSN2M8A46CgDV2H0o2H0qSigCPYfSjYfSpKKAI9h9KNh9KkooAj2H0o2H0qSigCPYfSjYfSpKKAKz2kUs8U0kEbyw58uRlBZMjBwe2RUuw+lZFjqGoN4w1LTLyS2ktYreG4t/KhZHQOzqVclyG+5nIC9enetugCPYfSjYfSpKKAI9h9KNh9KkooAj2H0o2H0qSigCPYfSjYfSpKKAI9h9Ky9W8L6Hr8kb67omnam8QIja8tY5igPUDcDitisTxRqGoaXY2t1p0lsF+2QxTpPCzl0kkVDtIddpG7OSGHHSgDTt7SK0to7e0gjggiUJHFEoVUUdAAOAKk2H0qSigCPYfSjYfSpKKAI9h9KNh9KkooAj2H0o2H0qSigCPYfSjYfSpKKAI9h9KjhtY7ZClvDHErMXKooUFick8dySSTTNWuLm00W9uLGHz7mKB3hi2lt7hSQMDk5PYVieC9bvdas5n1PULWe5j2CS2j0qewltiRnDxzSMxz2OADjjNAHRbD6UbD6VJRQBHsPpRsPpUlFAEew+lGw+lSUUAR7D6UbD6VJRQBHsPpRsPpUlFAGHL4O8OT6t/as3h7SpNR3iT7Y9lGZtw6NvxnIwOc1sbD6Vy+reItQsvF9rp1nfaTcmWSNW0hVZrzym+9OWD4RF64MZBx97LADrKAI9h9KNh9KkooAj2H0o2H0qSigCPYfSjYfSpKKAI9h9KNh9KkooAj2H0pk1slzA8NxEksUilXjdQysD1BB6ip6x/E+pPpWjm5TVdL0hQ6h73VeYYwf8AZ3puJPAG8dc84wQDTEe1QFUAAYAHal2H0rP8N6heap4ftrzUoFguJA24IpVXAYgOFPKhgAwByRnGTWpQBHsPpRsPpUlFAEew+lGw+lSUUAR7D6UbD6VJRQBHsPpRsPpUlFAEew+lUbDQdL0q4uZ9L0uzsprp99xJbW6RtM2ScsVA3HJPJ9a0W3BTsALY4BOATXO6ZqWsx+KDpOrz6dd+ZbNc/wCgwPG1oNwCq+5237sthsJnY3y+gBv7D6UbD6VJRQBHsPpRsPpUlFAEew+lGw+lSUUAR7D6UbD6VJRQBHsPpRsPpUlFAGJqXhDw7rF8LzV/D+l390AFE91ZxyuAOg3MCeK1hHtUBVAAGAB2rJ1nUNQsNd0WO3ktjZXtw1vPHJCxkz5buGVw4A+5ggqc56itugCPYfSjYfSpKKAI9h9KNh9KkooAj2H0o2H0qSigCPYfSjYfSpKKAI9h9KqalpFjrNk1nq9hbX9qxBaC6hWVCRyCVYEVfqC+FybCf7BLFDc7D5Uk0RkRW9SoZSR7Bh9aAKul6Hpmh2rW2iaZZ6dAzbzFaQJEpb1woAzV3YfSqHhvUZ9X8MadqF2I1nurZJZBEpChiMnAJJA/E1p0AR7D6U5ARnNOooAKKKKACiiigBHRZI2RxlWBBB7iuItfhpBbaDpmmNqcsv2G/N00zRDdNGcr5J54XZtTPPC11l9fQ6dYT3t7KY7e3jaWV8E7VUZJwOTx6VQ0bxRpuvSyxafJcrNEiyNFdWc1s+xs4YLKikqcHkAigBl74Z+2XmuT/a9n9rWKWe3ys+VtEg3dfm/1nTjp15qK68Irdzag0l4yrfaUmmkLHym3f84Oefv9Mdutb2T6mqTatCuvJpJEnnvbNchsDbtDBcZznOT6UAc3rHw+Ouf2ddaodCv9RtYfs8sl/oguIJI85+WJpcxsPXeRycg8Y2dY0TUr3TI9P0vVYbG1aD7NcB7ISFo8YJjwyhGxnkhlHHy1ci1iym1ibS4rjdeQRrLJGFb5VJwPmxjPtnPI9RVzJ9TQAWltHZWcNrACIoI1jQE5OAMD+VS1Fk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AQanpGm63Z/ZNZ0+11C23BvJu4FlTI6HawIzWb4W8I6b4Sjv00q1tbdLy7a4K21ssIUHG1ML1A5x9elaGoalb6XZPdXjusa4GI42kdieiqigsxPYAEmo9K1m01q1aewkkKo5R0mheGSNhztaNwGU4IOCBwQe9AGlRUWT6mjJ9TQBLRUWT6mjJ9TQBLRUWT6mjJ9TQBLRUWT6mjJ9TQBLRUWT6mjJ9TQBUi0vyvEdzqvnZ8+2it/K2/d2M7Zznvv6Y7VoVh2/inS7rVv7Ninn88syoz2sqRSsvVY5WUI5GDkKxPB9DWvk+poAloqLJ9TRk+poAloqLJ9TRk+poAloqLJ9TRk+poAloqLJ9TRk+poAlrP1vS/7Y09bXzvJ2zwzbtu7/AFcivjGR124/GreT6ms3WPEFjoUcbag1yfMzhba0luGAHVisasQo4yxAAyOeaANeiq1vcxXdtFcWsyywyoHjkRsqykZBB9KkyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AOnR5LeRIpTC7KQsigEofXB4NZGiaJd2WoXep6xfQ3uo3SJEz21sbeJY03FQELuc5ZiSWP4VqPJ5cbO24hQSdoJP4Acmqek6xa63p63unPK0LMyfvYXiZWUlWBRwGBBBGCBQBpUVFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AS0VFk+poyfU0AYmseH9Q1jUrcT6nANKhuI7r7ObLM4kQ7gFm34VcgfwFuWG7njoKzr7WLPTZ7WG9uPLkvJRFAm0sXY/QHA6cnjkc8irmT6mgCWiosn1NGT6mgCWiosn1NGT6mgCWiosn1NGT6mgCWiosn1NGT6mgCWqWqQajPagaPfQ2dyrZD3FsZ42HcMgdCfbDDnHUcGxk+pqjqus2ujW6zXrTkO21I7e3knkc/7McasxwOTgcDk0AHh3RhoGixWAmEzK7yO6xiNS7sXbag+6uWOBk4Hc9a06o6fqNtqljHeWM3mwSZw2CpBBwQQcEEEEEEAgjBqzk+poAloqLJ9TRk+poAloqLJ9TRk+poAloqLJ9TRk+poAloqLJ9TRk+poAfKHaJxE4RypCsVztPY471znhXQNa0LzE1PVdO1BJSZJZYdNeCeaU4+d3adweBjAUYAAGAAK6AsQMk4H1rJ0zxRpmr30lpZTz+fGu8LPbSw+Ymcb4y6gSLnHzJkcjnkUAbdFRZPqaMn1NAEtFRZPqaMn1NAEtFRZPqaMn1NAEtFRZPqaMn1NAEtFRZPqaMn1NAFTUtL/ALRvNNn87y/sFz9o27c7/wB26Y68ffznnpWhWJqfijS9Hu1t7+adWIDO8drLLHCD0MjopWMe7kDg+lawYkZByPrQBLRUWT6mjJ9TQBLRUWT6mjJ9TQBLRUWT6mjJ9TQBLRUWT6mjJ9TQBLSOu5GXpkYqPJ9TRuPr+tAFXQ9M/sbQbLTfN877LCsXmbdu/AxnGTj86v1nabq9nrEMs2m3HnxxStCzBSBuHXGRyORyOD2NXMn1NAEtFRZPqaehznNADqKKKACiiigDB8YQS3PgnWoLaJ5ppLGZEjjUszsUIAAHJNc7pWmatpl7eW1zLd319d6aF0/VZ4seUAv+pk8tQiEOdwO1SwODkrXe+X70eX70AeSeHtF1a28P62NNiMF9JYeVLb22izacXuAeXMskrCaUgsPMQ4PUt0rW0DT7Q+KTN4T0a40O1fSJI1mm0t7aNZzIvJicKSw6kkDd6nHHovl+9Hl+9AHB+HNE8QaV44K6hd2VzZppoUz2+myw+a5kYnLtM4L5+Y+ueg613NP8v3o8v3oAZRT/AC/ejy/egBlFP8v3o8v3oAZRT/L96PL96AGUU/y/ejy/egBlFP8AL96PL96AGUU/y/ejy/egBlFP8v3o8v3oAZRT/L96PL96AGUU/wAv3o8v3oAZRT/L96PL96AGUU/y/ejy/egDI8R6xJoehy3sFjc30qlUSG3gklYljjJVFZtozkkAnA6VU8I+TJp9xdJJeTXVzOZLqa70+WzLybQPljkVSEAAA68Dkk5NdF5fvR5fvQAyin+X70eX70AMop/l+9Hl+9ADKKf5fvR5fvQAyin+X70eX70AMop/l+9Hl+9AHE2OsLr/AIvT7fa6rapYzSLZW0uk3KRs4BUzPOY9nILBVDYwckkkBezp/l+9Hl+9ADKKf5fvR5fvQAyin+X70eX70AMop/l+9Hl+9ADKKf5fvR5fvQAyub8aTMmmpGLnWrVZNw3aVY/avNOP9W4WN3VT/eGz/fU4rp/L96PL96AMzw/HND4b06K6tY7OZLaNXt4vuxEKMqOTwOnU/U1oU/y/ejy/egBlFP8AL96PL96AGUU/y/ejy/egBlFP8v3o8v3oAZRT/L96PL96AGVieE7ea20u6S4ikiZtQunCupUlTM5B57EEEGt7y/ejy/egBlFP8v3o8v3oAZRT/L96PL96AGUU/wAv3o8v3oAZRT/L96PL96AGUU/y/ejy/egDz7xHoPib+3La9tNRsbiKbVLchDpUryW8SkkAuJwNoOSflHLZzXfU/wAv3o8v3oAZRT/L96PL96AGUU/y/ejy/egBlFP8v3o8v3oAZRT/AC/ejy/egBlYXizXbjQ9PhNnaXU81zL5SywWM10tuMEmR0iUsQMcDjJIGR1HQeX70eX70AY3hmK1h0CEWJu2R2d3kvbd4JZHZiXZkdVKksScbQPTitan+X70eX70AMop/l+9Hl+9ADKKf5fvR5fvQAyin+X70eX70AMop/l+9Hl+9ADCcKSATx0HeuL0A3Fz43muP+JreRLbSRy3Gq2Bt2tWLgiGIhEWRTg5YB/ur85GBXb+X70eX70AMop/l+9Hl+9ADKKf5fvR5fvQAyin+X70eX70AMop/l+9Hl+9ADKKf5fvR5fvQBxXizWfO1P+wLm21SHT5Yg13dWuk3FyJlYkeSjRxsq5x8zHkA4HJ3L2KBRGoQYUAYGMYFSeX70eX70AMop/l+9Hl+9ADKKf5fvR5fvQAyin+X70eX70AMop/l+9Hl+9ADKy/EWm3+raQ1ppl9BZSO6+Y89u0yOmfmQqrocHocMOM1r+X70eX70Acn4Gs9Ys49YXW3t2L6lIYvIsnt1ZcKNwDSPlTjjHoeTXU0/y/ejy/egBlPj70eX705V20ALRRRQAUUUUAY3jGeW18Ea1PbSvDNFYzOkkbFWRghIII5BrlPB989kdYuyNWt7axsY3n03Vb97u483aX81SzyBUZcAbXIJByFKnPeX1nbalp9xZXqCS3uI2ilTcRuVhgjI5HHpUB0bTjqEN95OLiGA2ysJGAaM/wsM4cem4HHbrQBgadrXiOG709dZ/s26TVoHktorOGSJrd1TeEZmdhICON+EwR93nipa+OL2Dwlqt9qsKNqenqpks1sp4HjZzhVKNuMig8CSMsr4O0cYrd0zwhoekXLzWdq5ZozEqz3Ms6QxnqkaSMViQ8fKgUYAGMAVJZ+F9Hsob2KO3eZL8bbj7VcSXBdcEBMyMxCDJwowoycAZoAz/AAvql74i0+/ttYS4HlsIxdR6fdaX5ysuTsSU+YpU5G4N6EEdBb8EySSeC9NaeaWd/KIMk0jSO2GI5ZiSTx1JzV3SNGstEt2hsTcMGIy91dy3LkDoN8rM2BzgZwMn1qxY2dtptlHaWSCKCIYRNxOOc9Tz3oAsUUmR6ijI9RQAtFJkeooyPUUALRSZHqKMj1FAC0UmR6ijI9RQAtFJkeooyPUUALRSZHqKMj1FAC0UmR6ijI9RQAtFJkeooyPUUALRSZHqKMj1FAC0UmR6ijI9RQAtFJkeooyPUUAUdbjtpNHuPt97LZWqrummhnMLBByfnGGX6qQfQis7wfDdRabcGU3os5Jy1jHqEryXCQ7R99nJfltzAOSwBAOOg0Na0aw8QacbHVFleAyJJiG4khYMpDKQ8bKwwQDwadpWlWuj2rQWkt3IjNuJu76a6bP+9K7MB7ZxQBeopMj1FGR6igBaKTI9RRkeooAWikyPUUZHqKAFopMj1FGR6igBaKTI9RRkeooA4pluLP4iWjZ1iziuJpEea8uzNbXv7slY4oldli24zuKxk7f4iSa7asiHwzpMGtf2qkUpuQWKCS6leKIsMMY4mYpGTzkqoJyfU1rZHqKAFopMj1FGR6igBaKTI9RRkeooAWikyPUUZHqKAFopMj1FGR6igBa4rxbd3Nz4lttJW11e7tVs2unttIufs0szbwozN5ke1VGTt3gsWHBxXaZHqKztX0LT9bWP7d9oV4s7JbW7ltpFB6jfEytg4GRnBwPSgBfD99BqOgWlzazTzRsmA1yAJcg4IfH8QIIP0rRqC0tbaws4rWzjSGCJdqRpwFFTZHqKAFopMj1FGR6igBaKTI9RRkeooAWikyPUUZHqKAFopMj1FGR6igCrq1vc3ei3tvYy+RcywOkMu4rscqQDkcjB7iua8DpJp095pGpQ3UWqW8UUsxk1i41GKVWBAdHmwVyVbK7R+PWusnihubeSC4VZIpFKujdGB6iqOkaFp2hxyiwWUvMQZJrm5kuJXxwAZJGZiB2BOB2oA0qKTI9RRkeooAWikyPUUZHqKAFopMj1FGR6igBaKTI9RRkeooAWikyPUUZHqKAOduUaH4j6e0dxchbmwuPNhNzIYiUaLaRGTsB+ZuQATnmujqtJZWsuoQ3zoDcwRvHG+4/KrFSwxnHO1fyqxkeooAWikyPUUZHqKAFopMj1FGR6igBaKTI9RRkeooAWikyPUUZHqKAFrC8U64+j2sCRQXzNdP5ZntNPmu/s6gZLFYkY57DIxk88A1uZHqKMj1FAHNfD3Uxqvgq0uBLeTEPInmXkcqyMBI2CfMAY8Y5/wrpqrafYWmlWMdnYRiK3jzsTcWxkknkknqTVjI9RQAtFJkeooyPUUALRSZHqKMj1FAC0UmR6ijI9RQAtFJkeooyPUUALXOeF0aDVvEVsLi5mihv18sXFzJMUDQxsQC5JAyxOBwM8Yrosj1FV7eytbS4up7dAkl3IJJm3E72ChQeTxwoHHpQBZopMj1FGR6igBaKTI9RRkeooAWikyPUUZHqKAFopMj1FGR6igBaKTI9RRkeooA53xEjR+I/DlxFcXMbNetA6R3MixyIYZGw0YO1uVBBIJGOK6Oq1zZWt5NbS3CB3tZfNhO4ja+0rng88MRz61YyPUUALRSZHqKMj1FAC0UmR6ijI9RQAtFJkeooyPUUALRSZHqKMj1FAC1R1qJp9FuY0jvZSU/1VhMIZpB3VHLLtJ6Z3KR2IPNXcj1FV7+yttSspLW7DGNxyY5GjdT2KupDKw7EEEdjQBg+BpJv7Pv4LiW7BgvGRbS+laae0UqpEbysW8w87shmHzYDECunqhpOj2GiWjW+nI6q7mSR5ZnmkkY9WaRyWY8AZJPAA7VeyPUUALRSZHqKXOelABRRRQAUUUUAZet6l/Y2gX+peV532O3efy923ftUnGcHHTrisrQvEl9fanPp+tafa2M8VrHdhrS9NzH5bkgBi0cZRvlPGCCOh4NaXiLTpdX8M6lp1syLNd2skKNISFDMpAzgE45rC0/wZ/Y63WnaSlra6LqNoUubaHMf2efZtLxKFxhh1HHIz3NAGnpnjHRNXnaKyupARGZla4tZYEljHV42kVRIo4+ZCRyDnkVNaeJtJvdLn1GK6ZLW3XfJJPC8OE6h8OASpwcMBg9iayNP0TX572w/ttdMhh0qF47drWWSU3TMmzc6MqiMbedgZ8k/e45pWvgnUJdB1rSrqWPT7S72Cztba/nuEgKnO5WYK0SkhR5SfKoX5Tk0AdJaa/Dq2lz3OiRS3E0R2fZ7mKS0cPwQGWRQyjBBzt6cgHpTPD2tXWrC+h1Cxjs7uxuPIlEFwZ4mO0NlHKoTwwBBUYOR71laJ4au7HRdRS60+zN1ehRJBc6nc6jHOFGNryzjcAQSMBSF6/N0q54Q0S70W2vBdw2ljHPMGg06wmeS3tEChdqFlXGSCxCooBPTOSQDoaKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigCC9kuYrGaSxhinuFQmOOaUxox9CwViB77T9Kr6FqZ1nQLHUmhEBuoFlMQfdsyM4zgZ+uBV513Rso6kEVneHdOl0jw1p+n3LI0trbpE7RklSQMcZA4oA0qKKKACiiigAooooAKKKKACiiigDJs9Xup/E2oaVdWcMSW0Mc8M0dwXMqOWHzKUGwgoeAW7c1rVmQ6ZNH4ru9TZo/Jms4YFUE7gyPISTxjHzjv61p0AFFFFABRRRQAUUUUAFFFFABWdrerro9g0qxG4uGVvJtw2DIQCTk4OFAGScce5wDo1z/ibwjZ+IopZnkvIr77K8EMlvqNxbKN3ZhE4DDOM5B6UAbGnXf2/S7W82eX9ohSXZnO3coOM9+tWKz9B0z+xvD1hpxdnNtAkbM8rykkDn5nJYjPTPatCgAooooAKKKKACiiigAooooAqatff2Xo15f8Al+b9lgeby923dtUnGecdKy/CHiK48TaV/aEi6T5DY8t9M1M3i5xyrHykCsOOOa3JxKbeT7MUWbadhkBKhu2QMHFYeg6VqMWs6hrGsRWdrc3qRRG2spmmQCPdhy7IhZjux90YAA5oA36KKKACiiigAooooAKKKKACiiigDJk1e6h8WwaTJZw/Zri1eaK5W4JfchUMpj2YA+cYO49+BWtWZc6ZNN4psNSVoxDbW08TqSdxLmMjHGMfIe/pWnQAUUUUAFFFFABRRRQAUUUUAFVr++i0+1M0oZiSEjjQZaRzwFUep/8ArnirNZms+HrDXjbnUDdq1sxaJ7S+mtmUkYPzROpPHHNADtA1R9a0OC/ltxbPLuDRCTeFKsVPzYGenoK0aw/B3h9/DPhmDTZpnmlRnZ3a5kmyWYnhpDnoRx65rcoAKKKKACiiigAooooAKKKKAEYlVJCliBkAdTWHp2t6k+uLputaXBZPPC89s1veGfKKygiQFF2N868AuOvzcDO3Jv8AKfyQpk2nYGOAT2z7VzHhTTtfs7ue48R2Gmfa7hcz39tqEk7yEH5UCNAgSMZOFDHHU5JLEA6miiigAooooAKKKKACiiigAooooAydU1e607WdKtls4ZbW/maB5jcFXifYzjCbCGBCnncMccGtaszVtMmv77SJoWjVbK88+QMTkr5TpgcdcsPTvWnQAUUUUAFFFFABRRRQAUUUUAFVtQmu4LJ3060W7ueBHFJN5SEnuzYJAHcgMfQGrNZfiOPWZtDlj8NtbLfOVVWuJTGqrn5iGCPhsZwdpGe1ACaFq9xqaXcV/aR2l5ZT+RPHDOZo921XBVyqkjDDqoOc8d61ayPDNlc6fpAtrvTbPTyjkiO0vHuQ+eS7O8aMWJJJJBJPJJzWvQAU+PvTKfH3oAfRRRQAUUUUAQ+Qf+e0n5L/AIUeQf8AntJ+S/4UG8tluhbNcQi4YZERcbyPp1qagCHyD/z2k/Jf8KPIP/PaT8l/wqaigCHyD/z2k/Jf8KPIP/PaT8l/wqaigCHyD/z2k/Jf8KPIP/PaT8l/wp0E8dzCJYG3ISQDgjocHr7ikiureeWSOGeOSSI4kRHBKH3HagBPIP8Az2k/Jf8ACjyD/wA9pPyX/CnTzw20JluZUijXq8jBQPxNKZohB5xkQRbd3mFhtx659KAGeQf+e0n5L/hR5B/57Sfkv+FOguILqIS2s0c0ZOA8bBh+YqSgCHyD/wA9pPyX/CjyD/z2k/Jf8Ka9/bR3q2jSHzmAO0ISAD0yQMDOOM9afNdW9sUFxPHEZDtQO4XcfQZ60AJ5B/57Sfkv+FHkH/ntJ+S/4UtxdW9pGJLueKBCcBpXCjPpk1IjrJGrxsHRhlWU5BHrQBF5B/57Sfkv+FHkH/ntJ+S/4VNSO4jRnc4VRkn2oAi8g/8APaT8l/wo8g/89pPyX/CnQTx3NvHPA26ORQytjGQfrUlAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4UeQf+e0n5L/hU1FAEPkH/ntJ+S/4U+OMpnLs+f72OPyFPooAKKKKACiiigDlLe5mtJ5YpPsk073xL2joTM4LcODnoFwR8pGB19NfxG8sejObeZ4JDLGokQ4Iy4FalFAHOXfk2l69vqGoXNvbx2wNu7XLqZHydx3Z+Zhx8vPXpUU0+otFa200s0U+owxjIJUxup+fH93KmuoooA5eO7n1DSJrxrxrZd0UeXdkXKj5wWHKAsSN3sK2tHuftWkwy4fnIBd9+cEjIbA3D0PcVeooAz9D/wCQRH/vyf8Aoxqbb/8AIy33/XCH+b1eggjtoRFAu1ASQMk9Tk9fc1JQBRuWuG0u8+1RRRkRPt8uQvkbT1yox+tZ9yD/AMI3pbsMwxtbvPx/AAMk+wOCfpW9RQBk6VIlxq2o3FoyvauYwrpyruAdxB79hn2qjqdyEvdQ+1Xs1vNGi/Yo0mZN/wAvUKOHJbgg5rpKKAMDT4S3ieeafzFm+yQs6+a20Md2RjOMe1Xdf/5Bi/8AXeH/ANGLWlRQBl6usUVxa3j3sVnJDvVHnTdGdw5B5GDxxyO/Wjw8kq6ODLn55HdAU2fKWJHHYd8e9alFAHN2N072c8i3Uz6wEl32xckKwzgeX0UdMEAZ9TmmW1wjPElhezXSvaubsSTNJsO3gnP3GzkY4+nFdPRQBzFle/Y00kzTNHA2mkBdxw7/AC4AHdvQdabaST3kWjJLdXIWW0keTZMylyNuMkc/1rqaKAOZ0WWf7RpLyXU8purWRpRJISCRtxx0GPXqe+abpzTiPTLlru5kkmupI3DzMVK/PgbenYc9feuoooAwdGuWGrT2z3LXTEM5cSltvzcBkI/dsM4AHBx0reoooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKAP/9k=)

Next a DiSTATIS Analysis will be done on this dataset.

DiSTATIS

## Data

We have the data related 18 wines, that have been assesed by 19 different experts. Wines tested basically are of 3 colors (Red, White, Rose).

TO find a general trend amongst the experts I randomly assign them gender (this was not given in data, any conclusion from this will not be significant).

Raw <- read.csv('19ExpertsFreeSort18Wines3Colors.csv', row.names=3)  
Sort <- Raw[,-c(1:2)]  
  
BoysGirls = c('f','m','f','f','m','m','m','m','f','m','m','f','f','m','m','m','m','f','m')  
Judges <- paste0(BoysGirls,1:length(BoysGirls))  
head(Sort)

## J1 J2 J3 J4 J5 J6 J7 J8 J9 J10 J11 J12 J13 J14 J15 J16 J17 J18 J19  
## Rose5 6 1 6 3 3 3 2 1 2 5 5 3 1 2 1 2 1 6 1  
## Red6 3 4 2 1 2 1 3 2 2 6 5 2 1 2 3 3 4 3 6  
## White5 4 1 1 5 3 3 2 1 1 4 5 2 4 3 2 1 1 1 1  
## Red4 2 2 3 1 4 2 3 2 4 7 4 2 6 4 5 2 4 2 6  
## Rose1 2 3 6 5 1 4 1 1 3 3 2 3 9 1 6 2 3 4 2  
## Rose3 1 1 4 5 1 4 2 1 2 5 5 1 2 3 1 4 3 6 5

#dim(Sort)

### Splitting into cubes based on the assessors (J1,J2,.. etc)

Each assessor will have a contingency table sort for the rows, where in the 2 or more rows having matching values will be 0, rest all will be 1. Similarly there will be cubes for all assessors.

### Distance Matrices

#---------------------------------------------------------------------  
# 2. Create the set of distance matrices  
# (one distance matrix per assessor)  
# (use the function DistanceFromSort)  
DistanceCube <- DistatisR::DistanceFromSort(Sort)

## Running DiSTATIS

#---------------------------------------------------------------------  
# 3. Call the DISTATIS routine with the cube of distance as parameter  
testDistatis <- DistatisR::distatis(DistanceCube)  
# The factor scores for the beers are in  
# testDistatis$res4Splus$F  
# the factor scores for the assessors are in (RV matrice)  
# testDistatis$res4Cmat$G

### **Bootstrap**

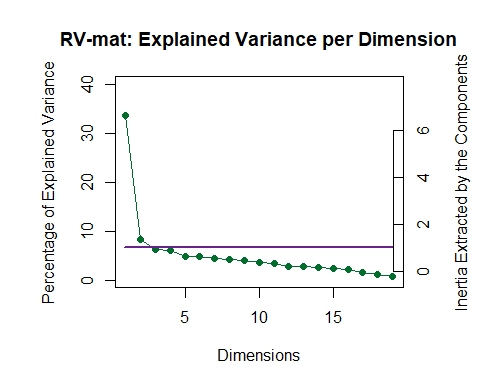
#---------------------------------------------------------------------  
# 4. Inferences on the beers obtained via bootstrap  
# 4.1 Get the bootstrap factor scores (with default 1000 iterations)  
BootF <- BootFactorScores(testDistatis$res4Splus$PartialF)

## [1] Bootstrap On Factor Scores. Iterations #:   
## [2] 1000

#  
#---------------------------------------------------------------------  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# End of computations  
# We have now in testDistatis  
#1 "$res4Cmat" "Results from the C matrix (see notation)"  
#2 "$res4Splus" "Results from the S+ matrix (see notation)"  
#

### **Scree Plot**

#---------------------------------------------------------------------  
# 5. The graphs  
# 5.1 First the C matrix  
# Look at the scree to start with  
#  
ev4C <- testDistatis$res4Cmat$eigValues  
Scree.1 <- PlotScree(ev = ev4C,  
 p.ev = NULL, max.ev = NULL, alpha = 0.05,  
 col.ns = "#006D2C", col.sig = "#54278F",  
 title = "RV-mat: Explained Variance per Dimension",plotKaiser = TRUE)



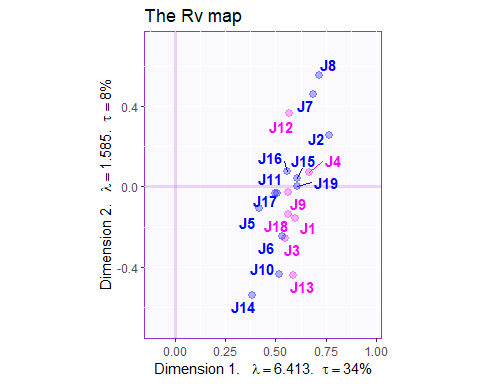
zeScree.Rv <- recordPlot()

### **Graph for the assessors**

**Note: Gender has been randomly assigned and plotted, results are not of any significance**

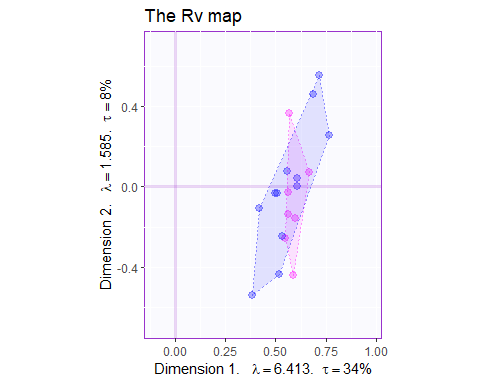
**Group Means and CI can’t be plotted for the assessors as they cannot be differentiated on any variables**

#---------------------------------------------------------------------  
#---------------------------------------------------------------------  
# A graph for the Judges set  
baseMap.j <- PTCA4CATA::createFactorMap(G,  
 title = 'The Rv map',  
 col.points = col4Judges,  
 alpha.points = .3,  
 col.labels = col4Judges  
 )  
# A graph for the J-set  
aggMap.j <- baseMap.j$zeMap\_background + # background layer  
 baseMap.j$zeMap\_dots + baseMap.j$zeMap\_text + expert\_labels\_12  
   
# We print this Map with the following code  
#dev.new()  
print(aggMap.j)



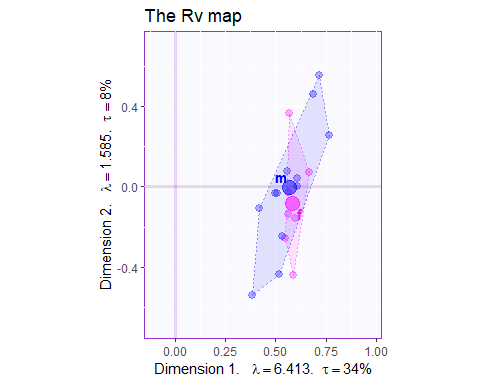
### **Hull for the expert’s plot**

GraphTJ.Hull.100 <- MakeToleranceIntervals(G,  
 as.factor(BoysGirls),  
 names.of.factors = c("Dim1","Dim2"),  
 col = unique(col4Judges),  
 line.size = .5,  
 line.type = 3,  
 alpha.ellipse = .1,  
 alpha.line = .4,  
 p.level = 1, # full Hulls  
 type = 'hull' #  
)  
#---------------------------------------------------------------------  
# Create the map  
aggMap.j.withHull <- baseMap.j$zeMap\_background + # background layer  
 baseMap.j$zeMap\_dots + GraphTJ.Hull.100 + expert\_labels\_12  
  
print(aggMap.j.withHull)



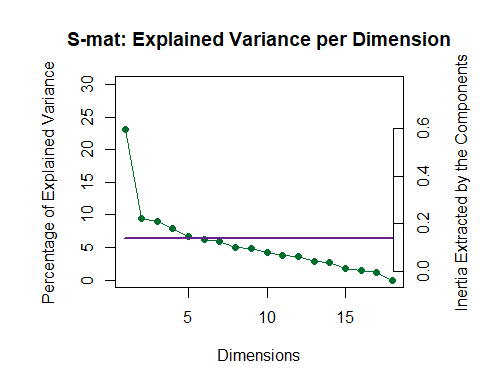
### **Mean**

#---------------------------------------------------------------------  
# Plot for the group means computed on the factor scores  
#---------------------------------------------------------------------  
# First compute the means  
JudgesMeans.tmp <- aggregate(G, list(BoysGirls), mean) # compute the means  
JudgesMeans <- JudgesMeans.tmp[,2:ncol(JudgesMeans.tmp )] # drop var 1  
rownames(JudgesMeans) <- JudgesMeans.tmp[,1] # use var 1 to name the groups  
#---------------------------------------------------------------------  
# a vector of color for the means  
col4Means <- unique(col4Judges)  
#---------------------------------------------------------------------  
# create the map for the means  
MapGroup <- PTCA4CATA::createFactorMap(JudgesMeans,  
 axis1 = 1, axis2 = 2,  
 constraints = baseMap.j$constraints,  
 title = NULL,  
 col.points = col4Means,  
 display.points = TRUE,  
 pch = 19, cex = 5,  
 display.labels = TRUE,  
 col.labels = col4Means,  
 text.cex = 4,  
 font.face = "bold",  
 font.family = "sans",  
 col.axes = "darkorchid",  
 alpha.axes = 0.2,  
 width.axes = 1.1,  
 col.background = adjustcolor("lavender",  
 alpha.f = 0.2),  
 force = 1, segment.size = 0)  
# The map with observations and group means  
aggMap.j.withMeans <- aggMap.j.withHull +  
 MapGroup$zeMap\_dots + MapGroup$zeMap\_text   
  
print(aggMap.j.withMeans)



### **For Rows**

# Go for the beers now  
# First we fix a bit of shamefull absentmindness:  
# The eigenvalues of the compromise matrix are not available  
# in DistatisR.  
# So we recompute them here  
ev4S <- eigen(testDistatis$res4Splus$Splus,  
 symmetric = TRUE, only.values = TRUE)$values  
# A scree for the compromise  
Scree.S <- PlotScree(ev = ev4S,  
 p.ev = NULL, max.ev = NULL, alpha = 0.05,  
 col.ns = "#006D2C", col.sig = "#54278F",  
 title = "S-mat: Explained Variance per Dimension",plotKaiser = TRUE)

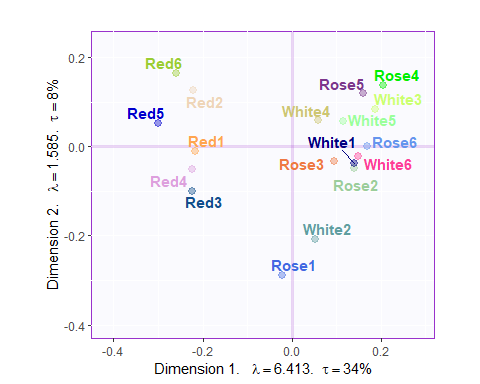


zeScree.S <- recordPlot()

Fi <- testDistatis$res4Splus$F  
col4Wines <- prettyGraphsColorSelection(nrow(Fi))

### **Graphs for I set**

# Graphs for the I set  
#---------------------------------------------------------------------  
# Create the base map  
constraints4Fi <- lapply(minmaxHelper(Fi),'\*',1.2)  
baseMap.i <- PTCA4CATA::createFactorMap(Fi,  
 col.points = col4Wines,  
 col.labels = col4Wines,  
 constraints = constraints4Fi,  
 alpha.points = .4)  
#---------------------------------------------------------------------  
# We are interested about the labels here  
# so we will use dots and labels  
#---------------------------------------------------------------------  
# Plain map with color for the I-set  
aggMap.i <- baseMap.i$zeMap\_background + baseMap.i$zeMap\_dots +  
 baseMap.i$zeMap\_text + expert\_labels\_12  
  
print(aggMap.i)



### **Confidence Interval**

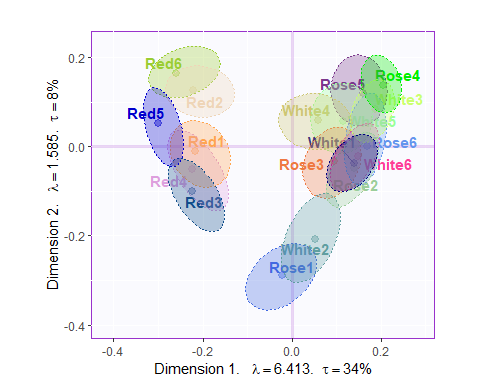
#---------------------------------------------------------------------  
# Create Confidence Interval Plots  
# use function MakeCIEllipses from package PTCA4CATA  
#  
constraints4Fi <- lapply(minmaxHelper(Fi),'\*',1.2)  
GraphElli <- MakeCIEllipses(BootF[,1:2,],  
 names.of.factors = c("Factor 1","Factor 2"),  
 alpha.line = .5,  
 alpha.ellipse = .3,  
 line.size = .5,  
 line.type = 3,  
 col = col4Wines,  
 p.level = .95 )  
#---------------------------------------------------------------------  
# create the I-map with Observations and their confidence intervals  
#  
aggMap.i.withCI <- aggMap.i + GraphElli + MapGroup$zeMap\_text + expert\_labels\_12  
  
print(aggMap.i.withCI)

## Warning: Removed 2 rows containing non-finite values (stat\_ellipse).

## Warning: Removed 13 rows containing non-finite values (stat\_ellipse).

## Warning: Removed 2 rows containing non-finite values (stat\_ellipse).

## Warning: Removed 2 rows containing missing values (geom\_text\_repel).

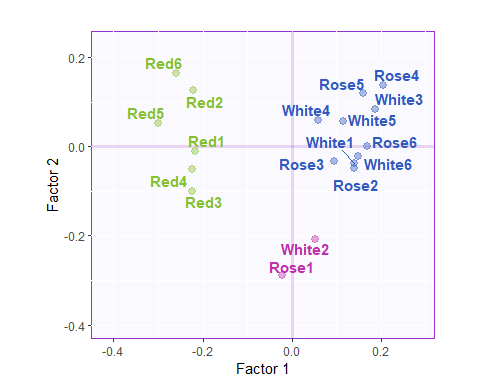


### **Partial Factor Scores**

# Old graph with links to partial factor scores  
# Not that informative for sorting tasks  
# Change names of the assessors  
partF <- testDistatis$res4Splus$PartialF  
dimnames(partF)[[3]] <- as.character(1:dim(partF)[3])  
PartialF <- GraphDistatisPartial(FS = testDistatis$res4Splus$F,  
 PartialFS = partF,  
 axis1 = 1, axis2 = 2, constraints = NULL,  
 item.colors = col4Wines,  
 participant.colors = NULL,  
 ZeTitle = "Distatis-Partial",  
 Ctr=NULL, color.by.observations = TRUE,  
 nude = FALSE, lines = TRUE)  
# save the graphs in F.and.PartialF  
F.and.PartialF <- recordPlot()

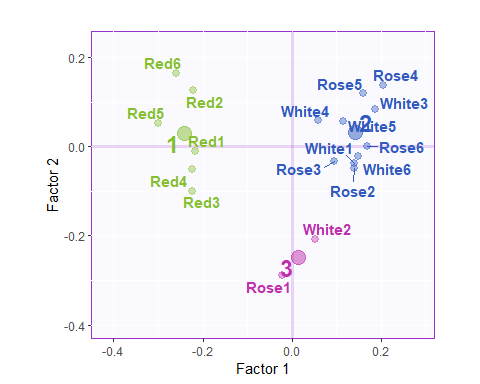
### **Graph fo rI set**

#=====================================================================  
#---------------------------------------------------------------------  
# Graphs for the I set  
#---------------------------------------------------------------------  
# Create the base map  
# constraints4Fi <- lapply(minmaxHelper(Fi),'\*',1.2)  
baseMap.i.km <- PTCA4CATA::createFactorMap(Fi,  
 col.points = col4Clusters$oc,  
 col.labels = col4Clusters$oc,  
 constraints = constraints4Fi,  
 alpha.points = .4)  
#---------------------------------------------------------------------  
# We are interested about the labels here  
# so we will use dots and labels  
#---------------------------------------------------------------------  
# Plain map with color for the I-set  
aggMap.i.km <- baseMap.i.km$zeMap\_background +  
 baseMap.i.km$zeMap\_dots + baseMap.i.km$zeMap\_text  
  
print(aggMap.i.km)



### **Adding cluster names**

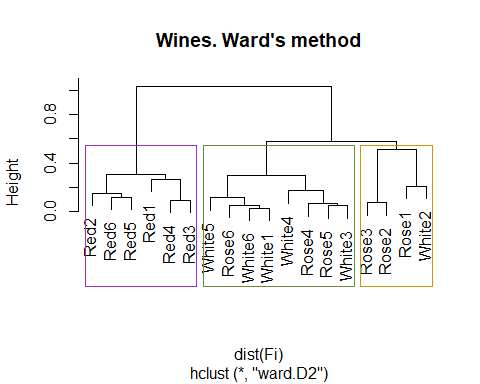
#---------------------------------------------------------------------  
# Add the cluster names  
#---------------------------------------------------------------------  
# get the color order in the c=good order  
col4C <- col4Clusters$gc[sort(rownames(col4Clusters$gc),  
 index.return = TRUE)$ix]  
# create the map for the means  
map4Clusters <- PTCA4CATA::createFactorMap(wines.kMeans$centers,  
 axis1 = 1, axis2 = 2,  
 constraints = constraints4Fi,  
 title = NULL,  
 col.points = col4C,  
 display.points = TRUE,  
 pch = 19, cex = 5,  
 display.labels = TRUE,  
 col.labels = col4C,  
 text.cex = 6,  
 font.face = "bold",  
 font.family = "sans",  
 col.axes = "darkorchid",  
 alpha.axes = 0.2,  
 width.axes = 1.1,  
 col.background =  
 adjustcolor("lavender", alpha.f = 0.2),  
 force = 1, segment.size = 0)  
# The map with observations and group means  
aggMap.i.withCenters <- aggMap.i.km +  
 map4Clusters$zeMap\_dots + map4Clusters$zeMap\_text  
  
print(aggMap.i.withCenters)



#---------------------------------------------------------------------

## Cluster analysis

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# A cluster analysis  
wine.hc <- hclust(d = dist(Fi),  
 method = 'ward.D2' )  
  
plot.tree <- plot(wine.hc, main = "Wines. Ward's method")  
hc.tree <- recordPlot()  
  
print(hc.tree)  
  
hc.3.cl <- rect.hclust(wine.hc, k = 3,  
 border = c('darkorchid',  
 'darkolivegreen4','darkgoldenrod3')  
 )



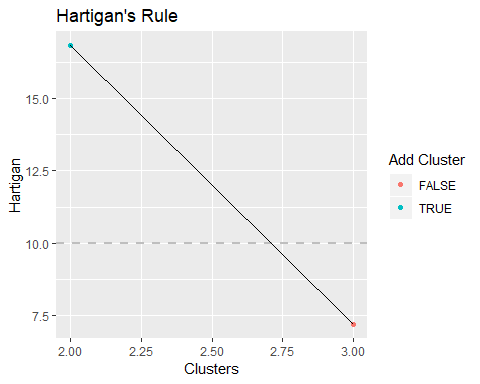
hc.tree.3c <- recordPlot()  
#dev.new()  
print(hc.tree.3c)

### **Optimal num of clusters**

# Optimal number of clustera  
# install.packages('useful')  
best.wines <- useful::FitKMeans(Fi, max.clusters = 3,  
 seed = 314)  
print(best.wines) # when Hartigan parameter > 10 => add a cluster

## Clusters Hartigan AddCluster  
## 1 2 16.85167 TRUE  
## 2 3 7.17265 FALSE

plot.harti <- useful::PlotHartigan(best.wines)  
print(plot.harti)



## Conclusion

* All of the Red Wines come under similar grouping.
* Rose 1 & White 2 together were the outliers to form the 3rd cluster.
* Remaining Rose & White wines were part of cluster 2.